

DEVELOPMENT OF A  
VISION-BASED LOCAL  
POSITIONING SYSTEM  
FOR WEED DETECTION

A Thesis Submitted to the  
College of Graduate Studies and Research  
in Partial Fulfillment of the Requirements  
for the Degree of Master of Science in the  
Department of Agricultural and Bioresource Engineering  
University of Saskatchewan  
Saskatoon

By  
Véronique Fontaine

## **PERMISSION TO USE**

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor who supervised my thesis work or, in his absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Agricultural and Bioresource Engineering  
University of Saskatchewan  
Saskatoon, Saskatchewan S7N 5A9

## **ABSTRACT**

Herbicides applications could possibly be reduced if targeted. Targeting the applications requires prior identification and quantification of the weed population. This task could possibly be done by a weed scout robot. The ability to position a camera over the inter-row space of densely seeded crops will help to simplify the task of automatically quantifying weed infestations. As part of the development of an autonomous weed scout, a vision-based local positioning system for weed detection has been developed and tested in a laboratory setting. Four Line-detection algorithms have been tested and a robotic positioning device, or XYZ $\theta$ -table, was developed and tested.

The Line-detection algorithms were based respectively on a stripe analysis, a blob analysis, a linear regression and the Hough Transform. The last two also included an edge-detection step. Images of parallel line patterns representing crop rows were collected at different angles, with and without weed-simulating noise. The images were processed by the four programs. The ability of the programs to determine the angle of the rows and the location of an inter-row space centreline was evaluated in a laboratory setting. All algorithms behaved approximately the same when determining the rows' angle in the noise-free images, with a mean error of  $0.5^\circ$ . In the same situation, all algorithms could find the centreline of an inter-row space within 2.7 mm. Generally, the mean errors increased when noise was added to the images, up to  $1.1^\circ$  and 8.5 mm for the Linear Regression algorithm. Specific dispersions of the weeds were identified as possible causes of increase of the error in noisy images. Because of its insensitivity to noise, the Stripe Analysis algorithm was considered the best overall. The fastest program was the Blob Analysis algorithm with a mean processing time of 0.35 s per image. Future work involves evaluation of the Line-detection algorithms with field images.

The XYZ $\theta$ -table consisted of rails allowing movement of a camera in the 3 orthogonal directions and of a rotational table that could rotate the camera about a vertical axis. The ability of the XYZ $\theta$ -table to accurately move the camera within the XY-space and rotate it at a desired angle was evaluated in a laboratory setting. The XYZ $\theta$ -table was able to move the camera within 7 mm of a target and to rotate it with a mean error of 0.07°. The positioning accuracy could be improved by simple mechanical modifications on the XYZ $\theta$ -table.

## ACKNOWLEDGEMENTS

I gratefully acknowledge the contribution of certain people to my personal and academic development:

- My supervisor, Dr. Trever G. Crowe, for teaching me the true meaning of “perfectionism”, and for allowing me to work on the most thrilling project possible.
- The members of my advisory committee, Dr. Martin Roberge, Dr. Huiqing Guo and Dr. Ron Bolton for their generous guidance.
- The staff of the Department of Agricultural and Bioresource Engineering, especially Louis Roth, Wayne Morley and Bill Crerar, from whom I learned so much during the construction of the XYZ $\theta$ -table.
- The Engineering Shop staff.
- The Department of Agricultural and Bioresource Engineering, NSERC, the College of Graduate Studies and Research, and The Harvey Scholarship for their essential financial support.
- Catherine Hui, for her impressive sense of organisation and devoted help.
- The warm and welcoming people of Saskatchewan for making my stay in Saskatoon an unforgettable experience.
- My family, Georgette, Roger, Yannie, Catherine, Ariane and all the others, for trying so hard to understand and remember what my project was about and for supporting me anyway.
- Finally, my love and best assistant, Jonathan, who more than anybody else had to put up with my ups and downs in those two years, and for being always encouraging and patient.

## **DEDICATION**

À mes parents, pour la fierté dans leurs yeux qui m'incite à me dépasser, à mes soeurs,  
pour être une source infinie d'affection, et à Jonathan qui, par son amour et sa présence,  
a fait de chaque jour de cet ouvrage un jour heureux.

## TABLE OF CONTENTS

PERMISSION TO USE .....	i
ABSTRACT .....	ii
ACKNOWLEDGEMENTS .....	iv
DEDICATION .....	v
TABLE OF CONTENTS .....	vi
LIST OF TABLES .....	viii
LIST OF FIGURES .....	ix
1. INTRODUCTION .....	1
2. LITERATURE REVIEW .....	5
2.1 Autonomous vehicles .....	5
2.2 Vehicle and implement guidance .....	6
2.2.1 Segmentation .....	7
2.2.2 Line detection .....	10
2.3 Robotics .....	13
3. OBJECTIVES .....	18
4. MATERIALS AND METHOD .....	20
4.1 Line-detection algorithms .....	20
4.1.1 Hardware and software .....	20
4.1.2 Description of the algorithms .....	24
4.1.3 Calibration of the testing set .....	33
4.1.4 Testing of the Line-detection algorithms .....	35
4.2 XYZ $\theta$ -table .....	37
4.2.1 Hardware and software .....	38
4.2.2 Calibration of the testing set .....	47
4.2.3 Testing of the XYZ $\theta$ -table .....	50
5. RESULTS AND DISCUSSION .....	55
5.1 Line-detection algorithms .....	55
5.1.1 Determination of stripe orientation .....	56
5.1.2 Determination of stripe location .....	65
5.1.3 Processing time .....	76
5.1.4 Summary of analysis and application of the algorithms .....	76
5.2 XYZ $\theta$ -table .....	78
5.2.1 Camera rotation .....	78

5.2.2 Camera positioning.....	80
6. CONCLUSION .....	86
6.1 Line-detection algorithms .....	86
6.2 XYZ $\theta$ -table .....	87
7. RECOMMENDATIONS AND FUTURE WORK.....	89
REFERENCES .....	91
APPENDIX A   LINE PATTERNS USED IN ALGORITHMS TESTING.....	94
APPENDIX B   LINE-DETECTION PROGRAM.....	96
APPENDIX C   EQUATIONS OF INTER-ROW SPACE CENTRELINES .....	97
APPENDIX D   DESCRIPTION OF THE XYZ $\theta$ -TABLE .....	101
D.1 Specifications of XYZ $\theta$ -table .....	101
D.2 Controls hardware .....	104
D.3 Controller program.....	117
APPENDIX E   DATA.....	121
E.1 Original and processed data .....	121
E.1.1 Line-detection algorithms data .....	121
E.1.2 XYZ $\theta$ -table data .....	122
E.2 Statistical analyses .....	123
E.2.1 Line-detection algorithms analyses .....	123
E.2.2 XYZ $\theta$ -table analyses .....	130
APPENDIX F   TOLERANCE OF Y-RAIL DISPLACEMENT .....	136



## LIST OF TABLES

Table 4.1 Factors describing the images collected in Line-detection algorithms testing. ....	36
Table 4.2 Factors describing the images collected in XYZ $\theta$ -table testing.....	51
Table 5.1 Error in the determination of the row angle (deg) in original set of data, $n = 288$ . ....	56
Table 5.2 Error in the determination of the row angle (deg), outliers removed.....	58
Table 5.3 Outliers removed from the data in row angle analysis.....	58
Table 5.4 Error in determination of the inter-row space location (mm) in original set of data, $n = 288$ . ....	65
Table 5.5 Error in the determination of the inter-row space location (mm), outliers removed. ....	67
Table 5.6 Outliers removed from the Stripe Analysis and Linear Regression algorithms data in inter-row space location analysis.....	68
Table 5.7 Outliers removed from the Hough Transform and Blob Analysis algorithms data in inter-row space location analysis.....	69
Table 5.8 Average processing time (s) per image for each algorithm, $n = 288$ . ....	76
Table D.1 Specifications of the motor on the X-rails.....	102
Table D.2 Specifications of the motor on the rails Y and Z, and on the $\theta$ -table.....	104
Table D.3 Specifications of the controller boards (Weeder Technologies, 2004). ....	108
Table D.4 Specifications of digital I/O board (Measurement Computing, 2004).....	111
Table D.5 Specifications of Hall switches (Allegro Microsystems Inc., 2004).....	116

## LIST OF FIGURES

Figure 1.1 Weed scout evaluating weed density at pre-determined locations in the field.....	2
Figure 1.2 Map quantifying weed density in the field.....	2
Figure 1.3 Field of cereal crop. ....	3
Figure 1.4 Representation of the orientation ( $\theta$ ) of crop rows and location of the inter-row space. ....	4
Figure 2.1 Range of the camera, view from the top. ....	13
Figure 2.2 Monorail (Lintech Company, 2004). ....	15
Figure 2.3 XYZ-table (Lintech Company, 2004).....	15
Figure 2.4 Rotational table (Lintech Company, 2004).....	16
Figure 4.1 Line pattern characteristics. ....	21
Figure 4.2 Line-detection algorithms testing set. ....	22
Figure 4.3 Graphical interface of the Line-detection program.....	24
Figure 4.4 Edge detection along the rows (a) and columns (b) of pixels.....	27
Figure 4.5 Normal parameterization of a line in the XY-space. ....	30
Figure 4.6 Parameterization of a line in the $p\theta$ -space. ....	30
Figure 4.7 Parallel lines parameterized in the XY-space (a) and in the $p\theta$ -space (b). ....	32
Figure 4.8 Angles marked on the periphery of the rotational platform.....	34
Figure 4.9 Calibration grid pattern. ....	34
Figure 4.10 Line pattern aligned with the grid pattern on the rotational platform. ....	35
Figure 4.11 Representation of the row orientation error and positional error.....	37
Figure 4.12 XYZ $\theta$ -table testing set. ....	38
Figure 4.13 $\theta$ -table. ....	40
Figure 4.14 Line pattern used in evaluation of the XYZ $\theta$ -table (dimensions in mm). ....	41
Figure 4.15 Stepper motor controller board (Weeder Technologies, 2004).....	42
Figure 4.16 Simplified block diagram of the control system of the XYZ $\theta$ -table. ....	43

Figure 4.17 Hall-effect switch and magnet. ....	44
Figure 4.18 XYZ $\theta$ -table Control window of the Controller program. ....	46
Figure 4.19 Home positions on rails X and Y. ....	47
Figure 4.20 Representation of the positional error and rotational error in images gathered for XYZ $\theta$ -table testing. ....	52
Figure 4.21 Representation of the coordinate systems of the XYZ $\theta$ -table and the images. ....	53
Figure 4.22 Calculation of the mean direction angle of the camera relative to a target. ....	54
Figure 5.1 Example of images collected to evaluate the Line-detection algorithms. Original image without noise (a), and with noise (b). ....	55
Figure 5.2 Outlier images in the determination of stripe orientation by the Blob Analysis algorithm. ....	60
Figure 5.3 Outlier image in the determination of stripe orientation by the Hough Transform and Linear Regression algorithms. ....	61
Figure 5.4 Outlier image in the determination of stripe orientation by the Linear Regression algorithm. ....	62
Figure 5.5 Example of cluster where “weedy centre points” didn’t contribute to the determination of the equation of the row with the Hough Transform. ....	63
Figure 5.6 Example of noisy image processed with the Stripe Analysis algorithm. ....	64
Figure 5.7 Outlier image in the determination of stripe location by the Blob Analysis algorithm. ....	70
Figure 5.8 Outlier image in the determination of stripe location by the Hough Transform and Linear Regression algorithms. ....	72
Figure 5.9 Outlier image in the determination of stripe location by the Hough Transform and Linear Regression algorithms. ....	73
Figure 5.10 Outlier image in the determination of stripe location by the Hough Transform and Linear Regression algorithms where the maximum number of 6 clusters was reached. ....	74
Figure 5.11 Outlier image in the determination of stripe location by the Linear Regression algorithm. ....	75
Figure 5.12 Example of image collected to evaluate the XYZ $\theta$ -table. ....	78
Figure 5.13 Camera rotation error by rotation angle and target location. ....	79
Figure 5.14 Error on camera rotation by rotation angle. ....	80
Figure 5.15 Error on camera positioning by angle of rotation and location. ....	81
Figure 5.16 Representation of the positional error by target location and	

rotation angle.....	82
Figure 5.17 Orientation of the Y-rail at different locations on the X-rails.....	84
Figure 5.18 Possible causes of positional error induced by the rotation of the camera. (a) camera tilted, (b) camera off centre.....	85
Figure A.1 Line patterns used in Line-detection algorithms testing. Row width = 51 mm, Row spacing (RS) = 152 mm. (a) Offset = 0, (b) Offset = $\frac{1}{4}$ of RS, (c) Offset = $\frac{1}{2}$ of RS, (d) Offset = $\frac{3}{4}$ of RS. ....	94
Figure A.2 Line patterns used in Line-detection algorithms testing. Row width = 51 mm, Row spacing (RS) = 229 mm. (a) Offset = 0, (b) Offset = $\frac{1}{4}$ of RS, (c) Offset = $\frac{1}{2}$ of RS, (d) Offset = $\frac{3}{4}$ of RS. ....	94
Figure A.3 Line patterns used in Line-detection algorithms testing. Row width = 76 mm, Row spacing (RS) = 152 mm. (a) Offset = 0, (b) Offset = $\frac{1}{4}$ of RS, (c) Offset = $\frac{1}{2}$ of RS, (d) Offset = $\frac{3}{4}$ of RS. ....	95
Figure A.4 Line patterns used in Line-detection algorithms testing. Row width = 76 mm, Row spacing (RS) = 229 mm. (a) Offset = 0, (b) Offset = $\frac{1}{4}$ of RS, (c) Offset = $\frac{1}{2}$ of RS, (d) Offset = $\frac{3}{4}$ of RS. ....	95
Figure C.1 Points forming the equations of the inter-row spaces in images characterized by a row spacing of 152 mm, an angle of $90^\circ$ and an offset of 0.....	97
Figure C.2 Rotation of a point about the centre of the image. ....	99
Figure D.1 XYZ $\theta$ -table and control system. ....	101
Figure D.2 Motor, power supply and ball screw driving the Y-rail on the X-rails. ....	102
Figure D.3 Motor actuating the belt of the Y-rail and gearbox.....	103
Figure D.4 Z-rail and $\theta$ -table.....	103
Figure D.5 Control system of the XYZ $\theta$ -table.....	104
Figure D.6 Screw terminal board. ....	105
Figure D.7 Simplified circuit (1 of 8) of the logic monitor board.....	106
Figure D.8 Picture of a logic monitor board.....	107
Figure D.9 Circuit of a controller board (Weeder Technologies, 2004). ....	109
Figure D.10 Digital I/O board (Measurement Computing, 2004).....	110
Figure D.11 Block diagram of the digital I/O board (Measurement Computing, 2004).....	112
Figure D.12 Pinout of the digital I/O board (Measurement Computing, 2004).....	113
Figure D.13 Hall-effect “home” switch on X-rail. ....	114
Figure D.14 Hall-effect “home” switch on Y-rail.....	114

Figure D.15 Hall-effect ‘home’ switches on $\theta$ -table. ....	115
Figure D.16 Close-up of a switch on the $\theta$ -table. ....	115
Figure D.17 Circuit of a Hall-effect switch (Allegro Microsystems Inc., 2004). ....	116
Figure D.18 Image Grabbing and Processing window. ....	117
Figure D.19 Image Display window. ....	118
Figure D.20 Manual Control window. ....	118
Figure D.21 Data Acquisition window. ....	119
Figure D.22 Actual Parameters window. ....	119
Figure D.23 Results window. ....	120
Figure F.1 Theoretical orientation of the Y-rail relative to the X-rails (view from top). ....	136
Figure F.2 Tolerance on displacement of Y-rail. ....	137

## **1. INTRODUCTION**

In recent decades, chemical application has become a major environmental issue in agriculture. In addition to their possible adverse effects on the environment, chemicals can represent a major economic input for producers. There is now a strong demand from the public and from the producers for reduced use of chemicals. Quantities of herbicides used could be reduced if applications were targeted. Generally, farmers spray herbicides uniformly over the field, without regard to the amount or location of the weeds. By targeting weed patches, herbicides would be applied only where they are needed, with the minimum amount required.

To allow targeted applications, weed infestations in the field must be quantified. There is currently no other way for a producer to do this than physically scouting each field and taking notes of the location of the weed infestations. With the recent advances in machine vision, this time-consuming weed scouting could possibly be done by a robot, or autonomous weed scout.

Palmer and Wild (2000) reported on the feasibility to build a motorised scouting robot for densely seeded crops (pulses, cereals, oilseeds). This robot would follow a predetermined path in the field and determine the weed density at some locations along that path using a machine vision system (Fig. 1.1). This geo-referenced information would then be put into a weed map (Fig. 1.2) and uploaded to an automatic sprayer. The weed scout could also be used for spot spraying, soil sampling and insect and disease detection.

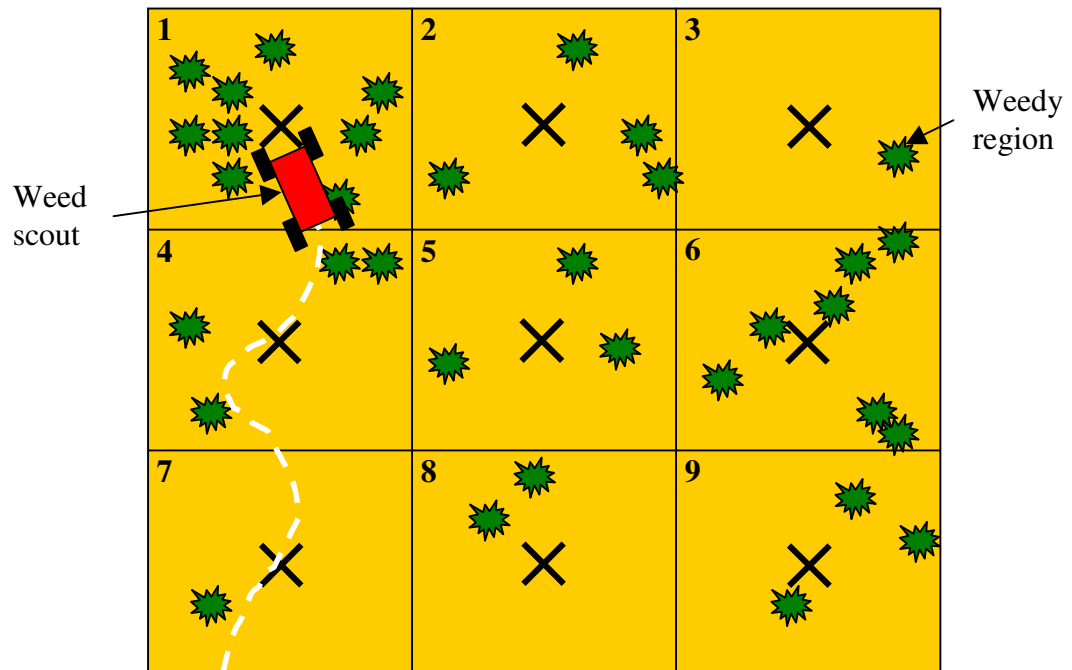


Figure 1.1 Weed scout evaluating weed density at pre-determined locations in the field.

1 <b>High</b>	2 <b>Medium</b>	3 <b>Low</b>
4 <b>Medium</b>	5 <b>Medium</b>	6 <b>High</b>
7 <b>Low</b>	8 <b>Low</b>	9 <b>Medium</b>

Figure 1.2 Map quantifying weed density in the field.

Under the assumption that the weeds growing within crop rows will eventually die because of the competition with the crop, the weed scout would be required to focus mainly on detecting weeds growing in the space between the rows, or inter-row spaces (Fig. 1.3). This assumption is often true in practice, but a number of factors influence the growth pattern of the weeds in the field (in the crop rows or in the inter-row spaces), for example moisture and nutrient applications. Cultural practices have also an effect on the weeds. For example, it is common for producers to do a pre-seed burnoff (complete eradication of the plant population with a non-selective herbicide) or to do a tillage operation prior to seeding. Those practices eliminate the weeds present in the field at the time of seeding, thereby facilitating the growth of the crop over the weeds within crop rows.



Figure 1.3 Field of cereal crop.

Because the detection of weeds would rely on image processing, the addition of a camera-positioning system would help to ensure good performance of the weed detection system. In order to determine the weed density at a specific location, the robot has to have the “best view” of the weeds, which is in an inter-row space (Fig. 1.4). Therefore, when arriving at a pre-determined location in the field, the robot would first



be required to identify the position and orientation of the rows in an image, then to position the camera of the vision system over the centreline of an inter-row space (Fig. 1.4). The present project's aim is to develop a vision-based local positioning system that could perform this task.



Figure 1.4 Representation of the orientation ( $\theta$ ) of crop rows and location of the inter-row space.

## **2. LITERATURE REVIEW**

In the literature, very few research projects are related to autonomous weed scouts. On the other hand, the problem of finding a linear pattern in an image was widely discussed. This present section is a review of research projects related to the present project.

### **2.1 Autonomous vehicles**

Different autonomous vehicles have been developed for agricultural applications. Researchers at the Danish Institute of Agricultural Sciences (Horsens, Denmark) and Aalborg University (Aalborg, Denmark) are currently developing an autonomous weed-mapping vehicle. The controls of this vehicle are widely explained by Pedersen et al. (2002), Sørensen et al. (2002) and Nielsen et al. (2002). The vehicle would travel from waypoint to waypoint and gather visual information at each waypoint. This information would yield weed maps used for spraying operations. Bak (2001) described the guidance system of this weed-mapping vehicle. The system consisted of a GPS receiver and a commercially available row guidance sensor. While the GPS receiver provided absolute positioning, the row guidance sensor modified the vehicle's trajectory to avoid causing damage to the crop. The guidance system was tested in a field of 0.25-m spaced winter cereal crop. However, Bak (2001) didn't mention how the row guidance system responded to the presence of weeds.

Åstrand and Baerveldt (1999a, 1999b, 2001, 2002, 2003) developed a row-following autonomous robot for the purpose of mechanical weed control. The robot described by Åstrand and Baerveldt (2001, 2002, 2003) was intended to work in crops sown in rows and presenting a high weed density. The robot followed a crop row and a mechanical tool removed the weeds growing in-between each plant of the row. The robot had two

vision systems: one for row following, and the other for plant identification. The row-following algorithm, described by Åstrand and Baerveldt (1999a, 1999b, 2001), was based on the Hough Transform (see Section 4.1.2). When tested with images of sugar beets and rape, the algorithm's offset error was 60 to 120 mm. In these images, the weed density was more than 3 weeds/crop plant, and the rows were widely spaced (0.48 m).

Researchers at the Silsoe Research Institute (Silsoe, UK) have been working for several years on an autonomous crop protection vehicle. This vehicle, mostly tested with cauliflower, followed crop rows and spot sprayed the plants individually. The row-tracking algorithm, developed by Marchant et al. (1997), was based on the Hough Transform (see Section 4.1.2). Southall et al. (2002) replaced this method by a crop grid pattern, which, superimposed on field images, provided localization as well as plant recognition information.

None of the above mentioned projects featured a positioning system for the camera. The camera was fixed to the autonomous vehicle. Therefore, positioning of the camera was achieved indirectly by positioning the vehicle.

## **2.2 Vehicle and implement guidance**

Positioning a camera over the space between two rows requires knowledge of the orientation of the rows ( $\theta$ ) and the location of a point on the centreline of the space between two crop rows, i.e. the inter-row space (Fig. 1.1). Different methods have been applied for row detection, mostly in vehicle guidance (Reid and Searcy, 1987a, 1987b; Marchant et al., 1997; Billingsley and Schoenfisch, 1997) and implement guidance (Slaughter et al., 1997; Tillett and Hague, 1999). In the case of vehicle guidance, the objective is to steer a vehicle with landmarks (crop rows or street lines) as guidelines, which imply real-time correction of the trajectory. In implement guidance, the objective is to control the lateral displacement of the implement with respect to crop rows.

Although research involving vehicle and implement guidance do not aim at the same

goal as research involving autonomous agricultural vehicles, it relies on the same steps of image processing. These steps are image recording, segmentation, line detection and extraction of the positioning parameters. The key steps are segmentation and line detection. The literature shows that different methods can be applied in each case.

### **2.2.1 Segmentation**

Once an image has been recorded, it is divided into objects and background. This is termed segmentation. It can be done by thresholding the image, edge detection or matching (Rosenfeld and Kak, 1976). For vehicle and implement guidance, the most common technique used is thresholding. The threshold is a value determined in the greyscale, which represents pixel intensity (Davies, 1997). Every pixel of the image below this threshold will take the lowest value of the greyscale (0), i.e. black, and every pixel above this threshold will take the higher value (255), i.e. white. This process is also equivalent to a binarization because only two greyscale values remain in the image. The resulting image is then ready to undergo line recognition using different statistical methods.

Discrimination between good plants, weeds and soil is a challenge because of the lighting variation throughout the day and growth season, and because of the angle of the sun, which causes shadowing. Images can also contain noise from straw, rocks and weeds in the inter-row spaces. A perfectly segmented image will be immune to non-plant objects. The goal of segmentation is to provide a partially processed image that includes only plants to the line detection system.

Plant reflectance is greater than soil reflectance in the near-infrared region of the light spectrum (Leamer et al., 1978). Marchant (1996), Marchant et al. (1997), Reid et al. (1985), Reid and Searcy (1986, 1987a, 1987b, 1988, 1991), Brandon and Searcy (1992) and Tillett and Hague (1999) all made use of this information and installed near-infrared filters on their cameras to enhance contrast between plants and background. Thus, the resulting infrared images were already partially segmented. This method has the

advantage of reducing the computational load of the segmentation process, thus allowing faster processing of the images. Processing speed is always an issue in vehicle or implement guidance because of its effect on guidance accuracy and travel speed. In this project, processing speed is not an issue because the camera will determine the positioning parameters from one picture only. The speed of each camera positioning operation will be influenced mostly by the speed of the displacement system. Nevertheless, because infrared images make the whole segmentation process simpler, it is a method that should be used in the development of the weed scout. Moreover, Brivot and Marchant (1996a, 1996b) proved that segmentation between crop and weeds is possible using near-infrared images and simple algorithms. This will be an asset when testing the prototype in the field, later in the development of the weed scout.

Instead of using near-infrared images, Søgaaard and Olsen (1999) accomplished their contrast enhancement with a combination (ratios, differences and multiplications) of the colour channels. The images were then thresholded and filtered so that small spots, which are likely to be weeds, would disappear. This allowed better row finding results. Their combinations worked well for their images in a wide range of lighting intensities. Some similar approach would be interesting to try in future developments of this current project.

The threshold value can be either calculated or manually set. The vision system developed by Billingsley and Schoenfisch (1997) for tractor guidance required the operator to set a threshold value. The operator was looking at a TV screen showing the segmented images. He could then manually modify the threshold to obtain the best segmentation with respect to natural lighting. Although this method gave very good results, it is obvious that it is of no help in the case of a weed scout robot because of the absence of any operator in the field.

The threshold can be calculated with a method called the Bayes classifier. Reid et al. (1985) and Reid and Searcy (1986, 1987a, 1987b, 1988, 1991) used this method after assuming that the grey level distributions of the canopy and soil background were

independent, normal and bimodal. Based on their image intensity distribution figures, this assumption seemed to be true, except in the case of images of plants 3 days after emergence. This could mean that their threshold calculation method should be used more carefully when working with something other than images of mature crops. It is likely to be the case in the present project because weed infestation problems occur mostly in the early stages of the plant growth.

Tillett and Hague (1999) also felt that Reid et al. (1985) and Reid and Searcy (1986, 1987a, 1987b, 1988, 1991) made an assumption that was false in some situations. Tillett and Hague (1999) stated that in practice, image intensity distribution was not always clearly bimodal. Unfortunately, they failed to provide any reference or evidence to support their claim. Nevertheless, Tillett and Hague (1999) rejected the Bayes classifier technique as a means to find the threshold value and calculated it by multiplying the average grey level in the picture by a manually adjustable factor. As mentioned earlier, this cannot be considered in the case of a weed scout because it would require an operator to go in the field and set this adjustable factor before every scouting mission. The weed scout would then rely on the operator's availability, which is undesirable for an autonomous system.

Slaughter et al. (1997) directly binarized their colour images by using a predefined lookup table that contained the two-class classification (crop and non-crop) for all colours possible in a 24-bit colour image. Each pixel was compared to the lookup table and then classified as crop or non-crop. Developing such a table for the present project would help reduce the processing time of segmentation because comparisons are faster than calculating the threshold value and applying it to each pixel. However, a lookup table is built relative to one single crop. Therefore, many tables would have to be developed to allow the weed scout to work in different crops and under different illumination conditions.

### 2.2.2 Line detection

When the objects or crop rows have been isolated, their pattern orientation has to be found. There are a number of statistical methods to do so, e.g. the Hough Transform (explained in Section 4.1.2) and regression analysis. In the field, line detection is complicated by the numerous uncertainties in the line pattern of the rows. Uncertainties can be caused by missing plants, variability in planting pattern, different plant sizes, and again, weeds, which usually do not disappear completely in the segmentation process.

In their research about the development of an autonomous agricultural vehicle, Marchant et al. (1997) overcame the uncertainty issue by using the Hough Transform, which they found to be a very robust method, tolerant of missing parts of the lines and presence of outliers, but only if the number of outliers was reasonably small compared with the number of crop points. To further increase the robustness of their system, Marchant et al. (1997) fused additional information from other instruments (odometer, accelerometer and compass) using a Kalman filter (a model that predicts the state of a time-controlled process) to determine their vehicle' s position with respect to rows. Their successful results definitely triggered an interest towards the Hough Transform as a line detection method for the prototype to be developed.

Reid and Searcy (1991) have put a lot of effort in developing an algorithm for line detection. They ended-up with a Heuristic line detection algorithm based on clustering. They applied this algorithm to their infrared images and used a run-length encoding procedure to find the middle of each row. The algorithm was based on the assumption that the rows were straight and required at least two rows in each image to be able to find the guidance parameters. They also evaluated the Hough Transform as another line detection method. Brandon and Searcy (1992) later extended the work of Reid and Searcy (1991) by developing a guidance algorithm considering curved row crop situations, as well as tillage and harvest operations, where only a single transition line existed in the image. Their work was based on the same techniques used by Reid and Searcy (1991). The assumption made by Reid and Searcy (1991) may not be always true

in a vehicle guidance situation because images cover a wide field of view, but it is a good assumption to make in the present project, because the camera's field of view will be less than  $1 \text{ m}^2$  (the rows will appear straight). Therefore, Brandon and Searcy's algorithm will probably not be considered here.

Søgaard and Olsen (1999) have been able to develop a method for detection and location of crop rows in small-grain crops (barley) using a technique very similar to the Hough Transform. They concentrated on detection of newly emerged barley crop rows under natural light conditions. These authors stated that the system gave good results even in the case of moderate weed infestation, although they failed to define or show a moderate weed infestation. The intriguing part of their line detection method lies in the kind of image they processed. The camera was looking directly downwards toward the crop so that the rows were parallel in the image. This is exactly what is required from a weed scout because looking downwards to the crop is the best way to see the weeds. All of the other authors cited (Reid et al., 1985; Reid and Searcy, 1986, 1987a, 1987b, 1988, 1991; Marchant, 1996; Marchant et al., 1997; Tillett and Hague, 1999; Slaughter et al., 1997; Billingsley and Schoenfisch, 1997 and Brandon and Searcy, 1992) used perspective images, i.e. rows pointing towards a vanishing point. Although sometimes very efficient, their algorithms cannot be applied in the case of a camera looking downwards because they take into account the camera angle and height and sometimes include a prediction of the vanishing point. The work of Søgaard and Olsen (1999) could be duplicated in this project. However, it would have been interesting to know the accuracy of their method before trying to duplicate it. Unfortunately, the project was still under development, and no results about the accuracy were presented. Nevertheless, their technique was one of the most interesting in the literature and it will certainly be considered in this project.

Billingsley and Schoenfisch (1997) used a technique very similar to regression analysis for line detection. Their system made use of the knowledge of the row spacing (the minimum distance from the centreline of a row to another). To minimize computational effort, only three viewports of the image were used for regression, each containing a



segment of a row. In the case of a weed scout, it would be possible to make use of additional knowledge like row spacing. This information could also be of great help in clustering methods.

Slaughter et al. (1997) eliminated the use of the Hough Transform because of its computational load in real-time control of a cultivator. Instead, they tested linear regression and three statistical estimates of central tendency, i.e. the mode, the mean and the median. Of the four methods, the median proved to be the most successful because of its lower sensitivity to outliers. Because their guidance accuracy results were as good as other Hough Transform-based methods, it would be interesting to investigate the efficiency of the median in the line detection program to be developed here.

Extraction of guidance parameters can be done with landmarks other than crop rows. Brown et al. (1990) tested three algorithms that processed digitized images of tilled and untilled soil and standing crop and stubble. Schönfeld and Pirsch (1993) developed a method consisting of many algorithms to extract guidance parameters from street lines. The process is similar to guidance of an agricultural vehicle: it implies image segmentation, binarization, and line detection with the Hough Transform.

No work has been reported on row detection in a closed canopy. The reason is believed to be that the methods cited previously are all based on plant/background differentiation, which is not possible with a closed canopy. However, Billingsley and Schoenfisch (1995) suggested that guidance in a closed canopy field could be made possible with the addition of tactile sensors to the vision guidance system. The same sensors could possibly be applied in future versions of the weed scout.

Most of the above-mentioned projects focussed on emerging crops. At emergence, the plants are still distant from each other within the same row. The smallest row spacing tested was 0.25 m, but cereal crops may be sown in rows 0.15 m apart, as in western Canada. It was acknowledged that images presenting a row spacing as narrow as 0.15 m should be processed in the present project.

All the projects cited above developed row-detection algorithms specifically for implementation in real-time guidance systems. The row guidance algorithms presented processed a sequence of images, and thus required high data-processing capabilities, but they were also simplified by allowing the assumption that rows were consistently oriented in the images. A system designed to process a single image from a given location, without the use of row following, must be capable of processing images of randomly oriented crop rows.

### 2.3 Robotics

In order to move the camera exactly over the centreline of an inter-row space, a robotic device has to be developed or bought. It seems there has never been any work done on positioning a stable device in 2D or 3D space with respect to crop rows. In the present project, the robotic device has to be able to displace a camera within a  $4 \text{ m}^2$  horizontal plane. If the field of view for the camera was less than  $1 \text{ m}^2$ , 4 different pictures could then be acquired at each location in the field (Fig. 2.1).

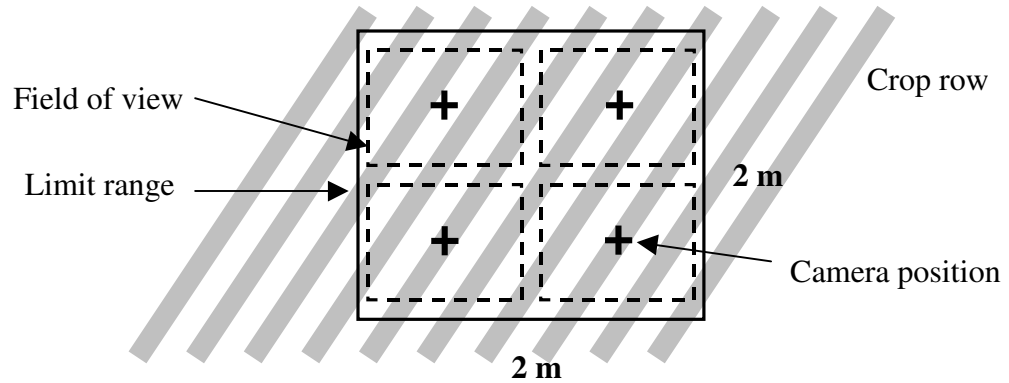


Figure 2.1 Range of the camera, view from the top.

The robotic device must be able to rotate the camera around a vertical axis (Z-axis) to position it in consistent orientations relative to the crop rows. This feature might be useful later in the development of the weed scout. For example, to determine the weed

density in a closed canopy, it may be necessary to introduce a device in the inter-row space that will “open” the canopy. One possible way to do this is to introduce two mechanical separators, or “arms”. If these separators are attached to the frame of the camera, they will be parallel to the rows after rotation. Once introduced in the inter-row space, they will move away from each other in a lateral displacement, opening the canopy at the same time and allowing the camera to acquire unobstructed views of the inter-row space.

Finally, to be able to position a canopy-opening system and insert it in an inter-row space, the robotic device is required to move an object up and down along the Z-axis over a range of approximately 25 cm. The Z-axis could also be used to position an illumination chamber that would provide artificial lighting to the crop, when weed scouting at night for example.

Two options were considered: a robotic arm and an XYZ $\theta$ -table. A robotic arm, though easy to control, is usually heavy and expensive. Moreover, it would be quite cumbersome to manipulate for the weed scout in the field.

An XYZ $\theta$ -table consists of rails allowing movement in each of the 3 orthogonal directions and rotation of the end-effector about the vertical axis. Figure 2.2 shows a monorail, and Figures 2.3 and 2.4 show examples of an XYZ-table and a rotational table, respectively. The XYZ $\theta$ -table could be built relatively easily, which was not the case with a robotic arm. Thus, the XYZ $\theta$ -table seemed the most appropriate for this project. No commercially available XYZ $\theta$ -table featured the dimensions required in this project, and custom fabrication or assembly was required.



Figure 2.2 Monorail (Lintech Company, 2004).

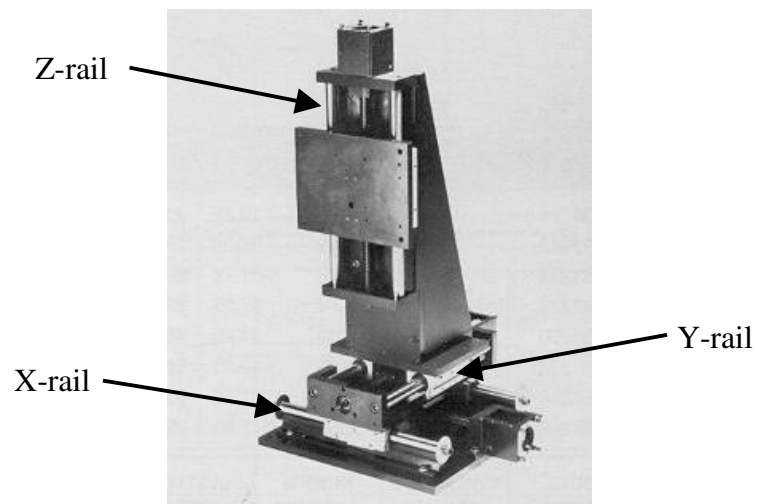


Figure 2.3 XYZ-table (Lintech Company, 2004).



Figure 2.4 Rotational table (Lintech Company, 2004).

One of the most important design considerations regarding XYZ $\theta$ -tables was the linear bearing system. It affects positional accuracy, load capacity and linear speed of the table (Lintech Company, 2004). The bearing system may be made of either steel balls recirculating or non-recirculating around a rod, or steel balls moving over and under linear rails.

Different motors may be used to induce motion on the rails. The stepper motor allows control of the distance travelled along the rail by knowing what distance is associated with one step of the motor. The other motors require sensors like limit switches and home switches to control motion along the rails. DC motors (or servomotors) have the advantage to be able to vary their speed. AC motors may also be used, but they are less likely to be considered because they do not present any interesting feature for this project like displacement and speed control. Moreover, it might be hard to have a power source for such motors in the field. The mechanical driving force may be transmitted to the system by lead screws or belts.

The controls of the XYZ $\theta$ -table should include data acquisition and control hardware such as motor driver boards (in the case of stepper motors), power supplies, a framegrabber, and possibly controller software. Often, a customized program is required to control the motors chosen and this program may be created using programming

languages such as Visual Basic or C++.

Knowing the exact position of the camera at all time, in a feedback configuration, is one of the challenges in the design of the XYZ $\theta$ -table. A system sensing the position along each rail could help avoiding situations like motor stalling, or data loss due to a cut of power. Relying on the control program to have the actual displacement is not sufficient. For example, if stepper motors are used, the control program will record the number of electrical pulses supplied to the motor, but if the torque required is too high, the platform will not move. Ideally, a sensing system along the rails will provide the actual position of the mobile element at all time. There are many ways to develop such a system, one being to install a series of sensors along the rails, and not only limit switches. Another way would be to use an ultrasonic device installed at the end of the rail, pointing towards the object moving on it. This device would emit pulses, which would be reflected from the object. The time required for these pulses to come back would indicate the distance between the device and the object. A third way of sensing position would be to create along the rails a system similar to a potentiometer. The length of the rail would be associated with a resistor, on which is applied a voltage. As the object moves along the rail, the voltage at each point is measured. Using the voltage divider law, which is ‘the voltage across each resistor in a series circuit is directly proportional to the ratio of its resistance to the total series resistance of the circuit’, it would be possible to deduce the position of the mobile element along the rail.

### **3. OBJECTIVES**

The over-arching goal of this project was to contribute to the development of an autonomous weed scout by developing a first prototype of the local camera-positioning system with respect to crop rows.

The specific objectives were:

- (1) to develop and test within a laboratory setting a machine-vision system capable of determining the position of a camera with respect to a set of artificial rows, and
- (2) to create and test a robotic prototype capable of accurately positioning a camera at a desired location and angular orientation.

In order to achieve the first objective, four different Line-detection algorithms were developed and tested. Each algorithm applied a different image processing or pattern recognition method. The algorithms were tested on the basis of:

- a) their ability to determine the angle of a set of parallel lines,
- b) their ability to determine the position (x,y) of a point on the centreline of an inter-row space, and
- c) their required processing effort.

In order to achieve the second objective, an XYZ $\theta$ -table was developed and tested on the basis of:

- a) its ability to accurately position an object within a horizontal plan, and
- b) its ability to accurately rotate an object about the vertical axis.



## **4. MATERIALS AND METHOD**

This section describes the Line-detection programs and the XYZ $\theta$ -table. The procedure followed to evaluate the programs and the XYZ $\theta$ -table is also described.

### **4.1 Line-detection algorithms**

The first specific objective of the present project was achieved by the creation of a Line-detection program including four Line-detection algorithms. The following section presents the Line-detection program in terms of hardware and software. Each Line-detection algorithm is described. An overview of the testing methodology and calibration is also included.

#### **4.1.1 Hardware and software**

Images were acquired using a digital colour video camera (2200 Series, Cohu Inc., San Diego, CA). The resolution of images was 640 x 480 pixels. The camera was equipped with a lens of 8.5-mm fixed focal length (Computar, Tokyo, Japan). The video signal was captured by a framegrabber (Meteor RGB, Matrox Electronics Ltd, Dorval, QC) installed in a desktop computer (Pentium III 833 MHz).

To simulate crop rows, different line patterns were built. A line pattern, illustrated in Figure 4.1, was a set of parallel thick black lines enclosed in a 1.19-m diameter circle with a white background. The black lines represented the crop rows, and the white spaces were the inter-row spaces. The patterns were plotted by a 1.22-m plotter. To facilitate handling, the patterns were glued on circular pieces of corrugated plastic board. To simulate a wide range of situations, the row width (RW) was either 51 or 76 mm, the

row spacing (RS) was 152 or 229 mm, and the offset from the centre was 0,  $\frac{1}{4}$ ,  $\frac{1}{2}$  or  $\frac{3}{4}$  of the RS value. The offset was the perpendicular distance from the centre of the circle to the middle of the nearest white row to the right-hand side (Fig. 4.1). A total of 16 different patterns were prepared. The patterns are presented in Appendix A. The circles were individually fixed with push pins on a rotary horizontal platform (Fig. 4.2) allowing the angular orientation of the rows to be controlled. Images of the lines were analyzed when the rows were oriented at 0, 30, 60, 90, 120, and 150° clockwise from the vertical orientation in the images. Uniform lighting was provided by four 500-W halogen lamps (Weatherproof Portable Halogen Work Light #60504, Globe Electric Company Inc., Pointe-Claire, QC ) equally disposed around the rotary platform.

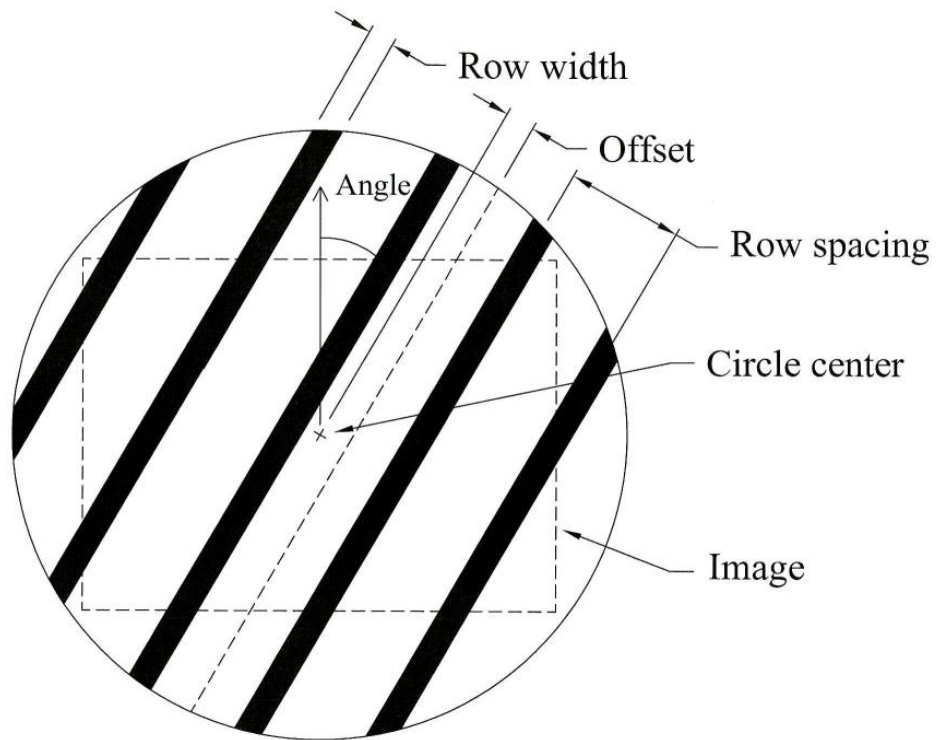


Figure 4.1 Line pattern characteristics.

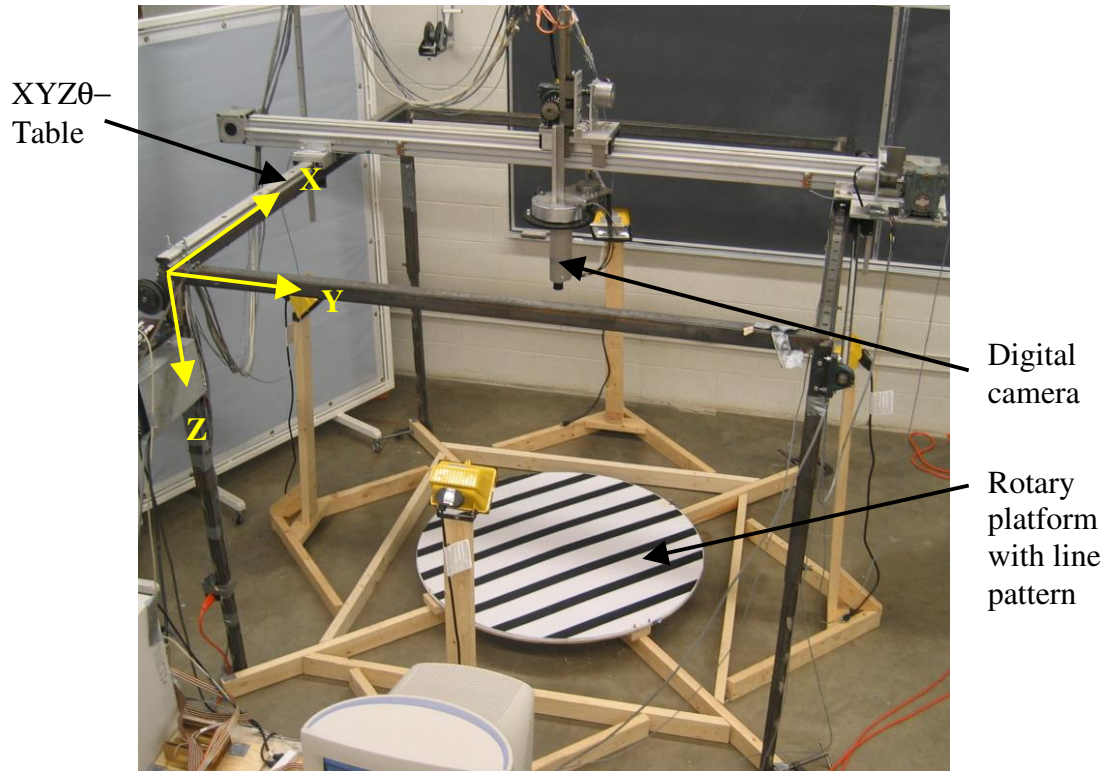


Figure 4.2 Line-detection algorithms testing set.

The camera was positioned over the platform, viewing downward. An XYZ $\theta$ -table (Fig. 4.2) was used as the camera stand. This table, described in Section 4.2, was able to move an object in the 3 orthogonal directions (X, Y, Z) and rotate it about the vertical axis ( $\theta$ -direction). The camera was positioned over the centre of the circles, at a height of 1.22 m. This resulted in an image size of 0.92 x 0.69 m, thus a resolution of 10 mm per 7 pixels.

Simulated weeds consisted of sunflower seeds (to simulate small, scattered weeds) and pieces of black foam cut in different shapes of 1700 mm<sup>2</sup> on average (to simulate weed patches). These were thrown randomly on the patterns. The overall surface area occupied by the "weeds" was approximately 0.05 m<sup>2</sup>. However, the actual area captured in the images was less than 0.05 m<sup>2</sup> because the foam and seeds may have coincided with the black lines or fallen outside the camera's field of view.

Four Line-detection algorithms were created, each one using a different image processing or pattern recognition technique. The algorithms utilized commands of a commercial imaging library (Matrox Imaging Library version 4.0, Matrox Electronics Ltd, Dorval, QC) for image capture, image processing and measurements. Microsoft Visual Basic 6.0 was the language chosen to implement the algorithms because of its compatibility with the imaging library, and because it allowed the development of a graphical interface.

The Line-detection algorithms were part of the same Visual Basic program. The program included an Image Collecting block and a Processing block. The Image Collecting block allowed images to be acquired and saved. Saved images were analyzed by each Line-detection algorithm included in the Processing block. The algorithms calculated the angle of the lines within the captured image and the location of a point on the centreline of an inter-row space. The angles were expressed as 0 to 180° relative to columns of pixels in the image, with angles increasing clockwise. The location of the inter-row space centreline was expressed in terms of direction (0 to 360° from Y-axis) and distance (in pixels) of displacement from the centre point of the image (320, 240) to a point on the inter-row centreline. Results for each algorithm were presented in a Results window. Another window (Actual Parameters window) allowed the creation of filenames for the results based on the actual parameters describing the image processed. The Line-detection program is presented in Appendix B. Figure 4.3 presents the graphical interface of the Line-detection program.

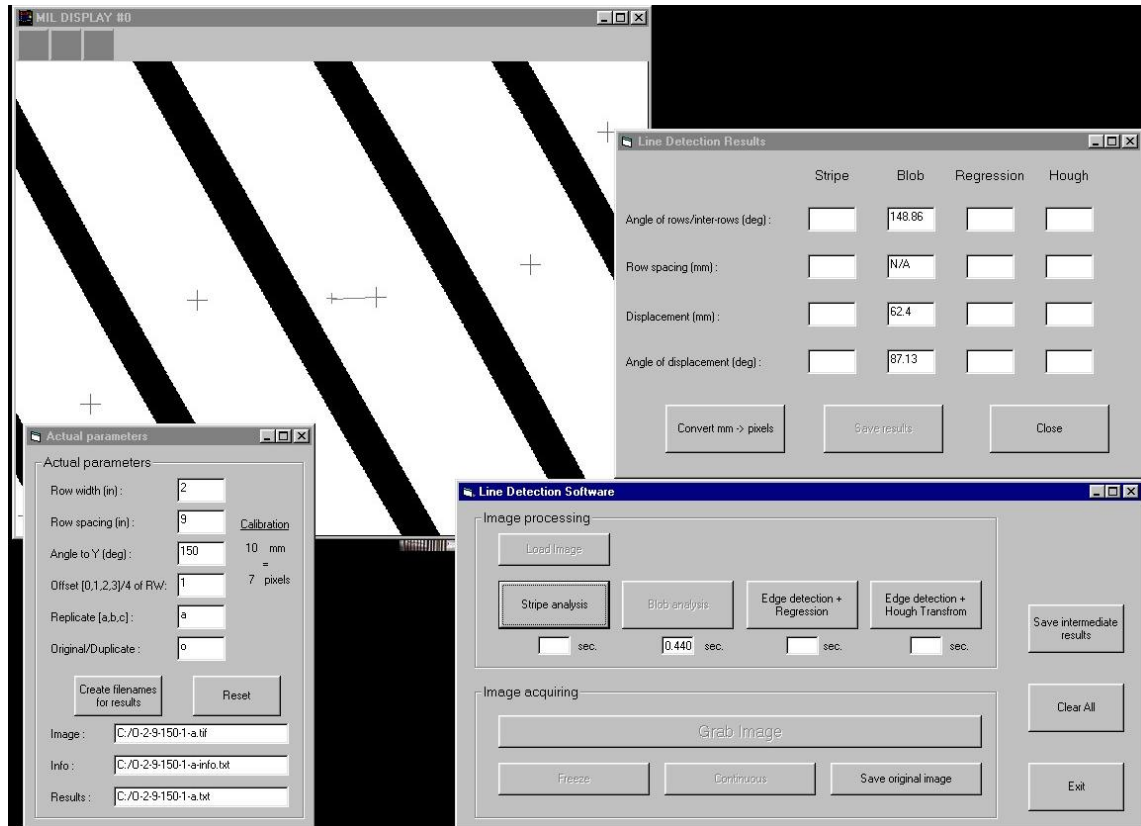


Figure 4.3 Graphical interface of the Line-detection program.

#### 4.1.2 Description of the algorithms

The measurement module of the imaging library contained a set of commands allowing features within an image to be characterized. One of the features that can be sought is a stripe, i.e. a region bounded by two edges. The Stripe Analysis algorithm consisted of commands to find a white stripe feature (i.e. an inter-row space) in the image processed.

The algorithm first converted the colour image into a greyscale image. The algorithm required *a priori* definition of the approximate width of the white stripe, and the region of the image within which the search should be conducted (i.e. the ‘search box’). In all cases, the stripe width was defined as 90 pixels (129 mm) with a variation of 40 pixels (57 mm). The search box was square, 330 pixels by 330 pixels, and centered in the image. The algorithm searched for a white stripe that was parallel with the vertical edges of the search box. The search box was rotated about its centre in increments of 4°,

within a specified range of 0 to 180°. At each increment angle, the search was performed again and a confidence score was calculated. The confidence score was the probability that the content of the search box corresponded to a white vertical stripe. The score was calculated based on the geometric constraints specified in the program, the stripe width for example. The angle presenting the highest confidence score was used for a finer search by step increments of 0.2°. After the stripe feature was located, the algorithm determined the position of its centre, which allowed calculation of the distance from the stripe to the centre of the image. The orientation of the white stripe was given by the angle of the search box presenting the highest confidence score.

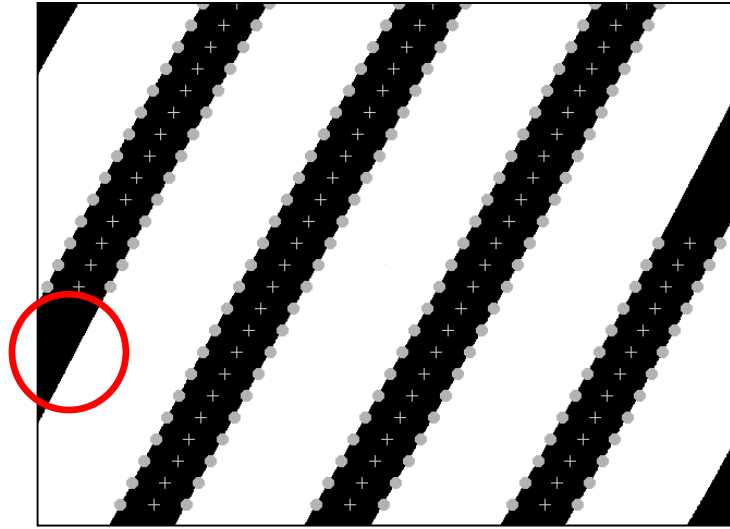
The Blob Analysis algorithm used another set of commands of the imaging library: the blob analysis commands. A blob analysis is an image processing operation with the goal to find and characterize regions of contiguous pixels of the same greyscale value in a binarized image.

First the colour image was converted to greyscale, then binarized. The algorithm searched for white blobs (inter-row spaces) of more than 200 pixels, as smaller blobs could represent noise. Once the blobs were found, the algorithm determined the angles of principal axes and centres of gravity. For perfectly straight white stripes, the centre of gravity was over the centreline of the white stripes. The algorithm then selected the blob whose centre was closest to the centre of the image and determined the number of pixels to the blob's centre of gravity. Thus, the direction and distance from the centre of the image to the centre of the nearest stripe was determined.

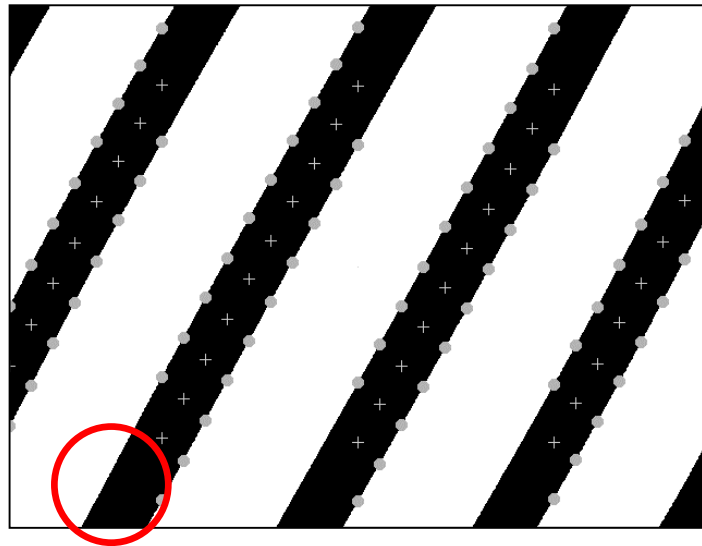
It was expected that this algorithm would provide accurate results as long as the two main hypotheses were true, i.e. that the centre of the closest blob was over the centreline of the inter-row space, and that the angle of this blob was representative of the angle of the crop rows.

The last two algorithms (Linear Regression and Hough Transform) shared a series of image processing steps. First the algorithms performed the same greyscale conversion

and binarization steps as the Blob Analysis algorithm, using commands from the imaging library. Edge detection was conducted by searching for pixel value transitions (255 to 0 or 0 to 255) along rows and columns of pixels. The rows and columns used in the search were separated by 19 pixels to reduce computational requirements and volume of information. The centre points of black stripes were then defined as the points midway between successive 255-0 and 0-255 transitions. Figure 4.4 presents an example of the edge detection along the rows and columns of pixels. If only one transition was found, no centre point was defined. This situation occurred in images where the stripes were at 30, 60, 120 and 150°. In those cases, some stripes had an edge partially outside the camera's field of view (Fig. 4.4).



(a)



(b)

• = stripe edge  
 + = centre point of black stripe  
 ○ = portion of the "crop row" where only one edge was present in the image

Figure 4.4 Edge detection along the rows (a) and columns (b) of pixels.

In cases when the stripes in the image were either horizontal or vertical, there would not



be any white-to-black or black-to-white transitions when the edge detection was performed on the rows and columns of pixels, respectively. To overcome these cases and improve the performance of the algorithm in other cases, the number of centre points found after completing each centre point search was recorded. The search that resulted in the greatest number of centre points was used in subsequent analysis.

There was the possibility that, in noisy images, the points found would correspond with the centre of simulated weed patches. It was acknowledged that this might affect the performance of the algorithm, both in the case of the Linear Regression and the Hough Transform algorithms. To minimize this effect, the edge detection step ignored sequences of successive transitions that were separated by less than 15 pixels (21 mm), because they were most likely patches of simulated weeds.

The next step was to define the stripes of which the points were members. This was done with a clustering technique termed the Similarity Matrix (Meisel, 1972). This method, similar to the Nearest-Neighbour method (Meisel, 1972), was also applied in the Hough Transform algorithm. This matrix defined the neighbours of each centre based on the Euclidean distance. The neighbour of a point A was defined as a point B whose location was within a certain distance from A. That distance, or threshold, was 40 pixels (57 mm). A first group of neighbours was formed, which was then merged with a second group of neighbours, etc., until no other neighbour surrounding the cluster could be found. That cluster corresponded to a crop row. Then a new similarity matrix was defined with the remaining points. A second cluster was formed, and so on. Two events could stop the clustering procedure: if all points were already clustered, or if 6 clusters were defined, which was considered enough to extract the positioning and orientation parameters. Only clusters of more than 9 centre points were considered in the subsequent processing steps. Smaller groups may represent simulated weed patches or portions of a row, which could lead to false results.

The Linear Regression algorithm performed a linear regression on each group of centre points to determine the equation describing each row. The average angle of the rows was

calculated from the slope of the equations. The equations of the 2 row centrelines closest to the image centre were selected. Using these 2 equations, the location of the nearest inter-row space centreline was defined as being midway between the two row lines, allowing the distance and direction to a point on the inter-row space centreline to be specified. The direction of that point was defined as being perpendicular to the orientation of the rows. Therefore, the point described on the centreline was the one closest to the centre of the image in ideal conditions.

With a conventional linear regression, it was impossible to describe a vertical line because its slope was infinite. In such cases, the dependent and independent variables in the regression were swapped. The algorithm decided which regression should be performed based on what centre point search (along the rows or columns of pixels) provided the highest number of centre points, which was an indication of the angle of the crop rows.

The Hough Transform algorithm called the same subroutines of greyscale conversion, binarization, edge detection and clustering as the Linear Regression algorithm. However, instead of performing a linear regression on each group of centre points, this algorithm applied a method called the Hough Transform.

The method consisted of calculating every possible line passing through every centre point of one of the clusters found. These equations were put into an accumulator and the most ‘popular’ equation was the one passing through the greatest number of centre points of the clusters. The conventional way of describing a line with a slope and an intercept implied the possibility of infinite slope. For this reason, Duda and Hart (1972) suggested to parameterize the line in the normal space  $(\rho, \theta)$  when applying the Hough Transform. The parameterization is given by:

$$\rho = x \cos \theta + y \sin \theta \quad (4.1)$$

The parameters of equation (4.1) are represented in Figure 4.5.

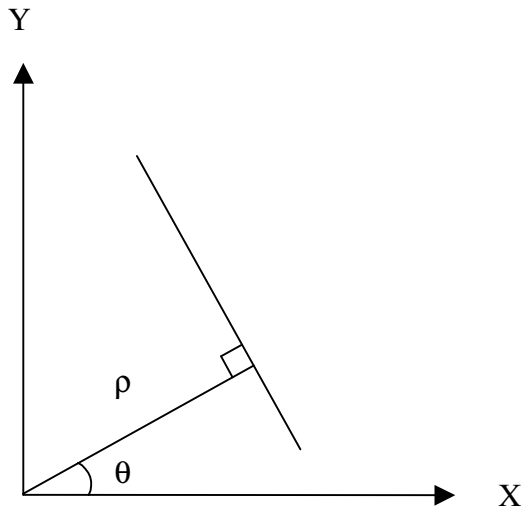


Figure 4.5 Normal parameterization of a line in the XY-space.

With this parameterization, a point in the XY-space was described as a sinusoidal curve in the  $\rho\theta$ -space, and all the points included in the same line in the XY-space were described as a set of sinusoidal curves intersecting at one point  $(\rho, \theta)$  in the  $\rho\theta$ -space (Fig. 4.6).

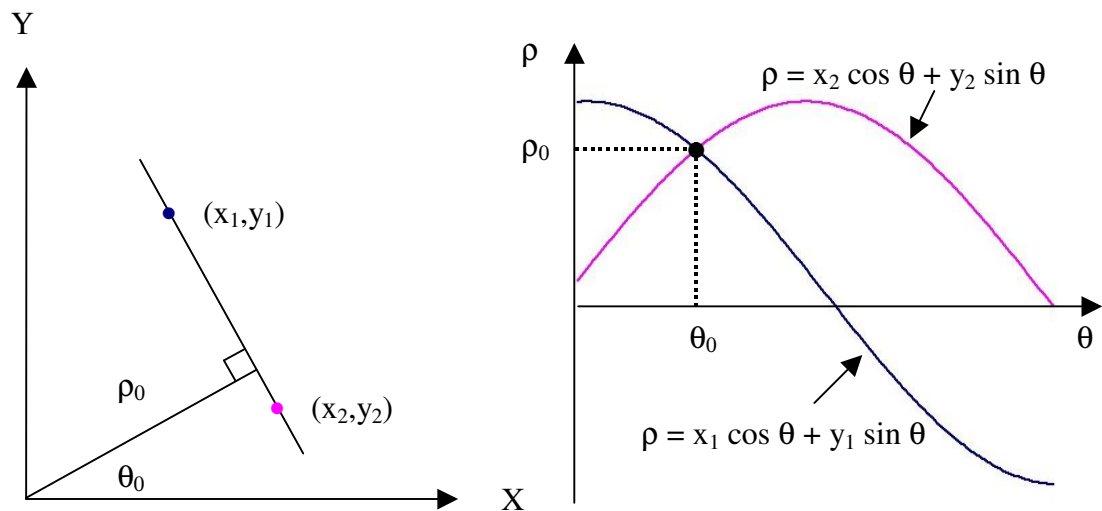
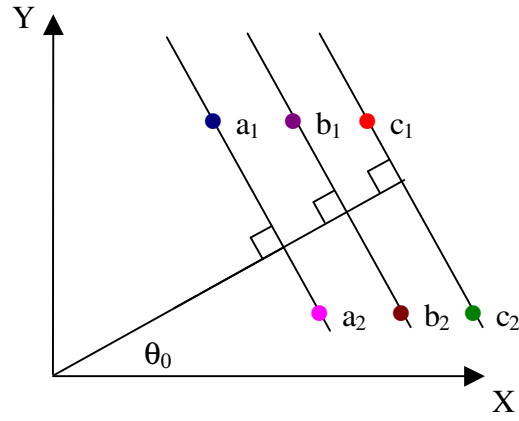


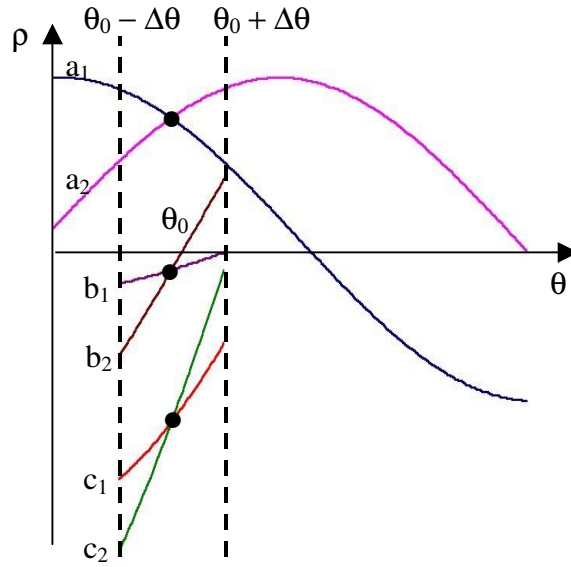
Figure 4.6 Parameterization of a line in the  $\rho\theta$ -space.

From the coordinates  $(x,y)$  of the centre points included in a cluster, sinusoidal curves in the  $\rho\theta$ -space were defined and put in an accumulator. The accumulator was defined as a three-dimensional matrix. The first dimension represented the axis  $\theta$  (from 0 to 180), the second represented the axis  $\rho$ , and the third dimension was the number of sinusoidal curves ( $n$ ) passing through each point  $(\rho,\theta)$ . The peak value of  $n$  corresponded to the intersection  $(\rho_0,\theta_0)$  of the sinusoidal curves (Fig. 4.6). From  $(\rho_0,\theta_0)$  and equation (4.1), it was possible to find the equation of the line passing through the greatest number of centre points in a cluster describing a “crop row”. Only clusters where at least 3 centre points contributed to the peak were considered in the subsequent analysis. The excluded clusters may not present a longitudinal profile and may represent simulated weed patches.

The sinusoidal curves of the first cluster were defined in the  $\rho\theta$ -space for values of  $\theta$  ranging from 0 to 180°, by increments of 1°. The intersection  $(\rho_0,\theta_0)$  of the first cluster was found. The parameterization presented in equation (4.1) implied that a set of parallel lines in the XY-space was defined as several groups of sinusoidal curves intersecting at different values of  $\rho$ , but at the same value of  $\theta$  (Fig. 4.7). Therefore, to reduce computational requirements, the search for an intersection  $(\rho,\theta)$  was narrowed to  $\theta_0 \pm 10^\circ$  in the subsequent clusters analyzed (Fig. 4.7).



(a)



(b)

Figure 4.7 Parallel lines parameterized in the XY-space (a) and in the  $\rho\theta$ -space (b).

Once an equation was found for each stripe, the average angle of all stripes was calculated from the average slope. The equations found also allowed the selection of the 2 rows closest to the image centre, which allowed the determination of the nearest inter-row space location. The direction and perpendicular distance from the image centre to the centreline of that inter-row space were calculated.

#### **4.1.3 Calibration of the testing set**

To ensure that the whole image would be included in the 1.19-m diameter line patterns, the camera height was adjusted to obtain a resolution of 10 mm: 7 pixels. This was accomplished with the help of the vision system. A black sheet of paper of known dimensions was placed in the camera's field of view. Images of the sheet were gathered at different angles and converted to greyscale. They were then enlarged using image processing software (Matrox Inspector 2.0, Matrox Electronics Ltd, Dorval, QC) until the pixels could be distinguished. Four line markers were superimposed on what was considered to be the edges of the black sheet. To avoid false results due to shadows around the edges of the sheet, the first and last pixel chosen for each line had approximately the same greyscale value. The software provided the length of the line markers in pixels. If the distance:pixel count ratio of the sheet edges was larger than 10:7, the camera was lowered; if it was smaller than 10:7, the camera was raised.

The angle positions on the rotating platform were determined using trigonometry and a 60-mm compass. Each angle was marked on the periphery of the rotating platform (Fig. 4.8). The orientation of the camera was adjusted to have the stripes vertical in the image when the pattern was at 0°. To determine the right camera orientation, a 1.19-m diameter grid pattern (Fig. 4.9) was plotted and fixed on the rotating part of the platform, with its centre over the centre of rotation of the platform. On the grid pattern, a cross was marking the angles 0, 90, 180 and 270°. The centre of that cross corresponded to the centre of the circular grid pattern. When installing the grid pattern on the rotating platform, the angles marked by the central cross were positioned according to the angles marked on the platform (Fig. 4.8).

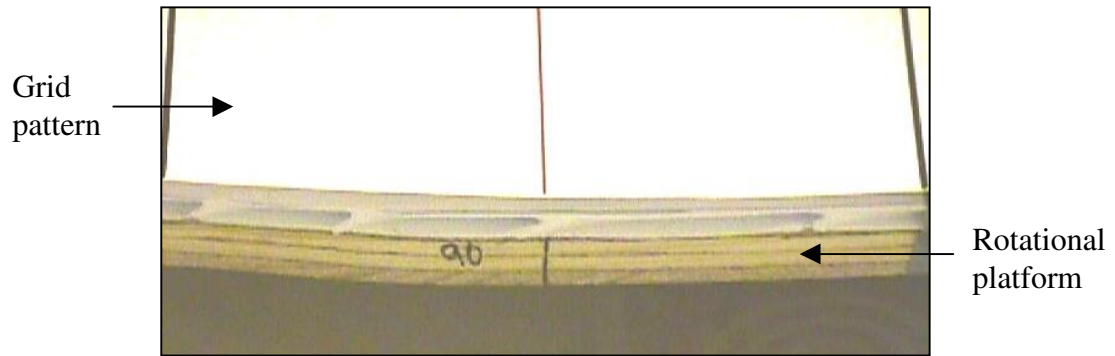


Figure 4.8 Angles marked on the periphery of the rotational platform.

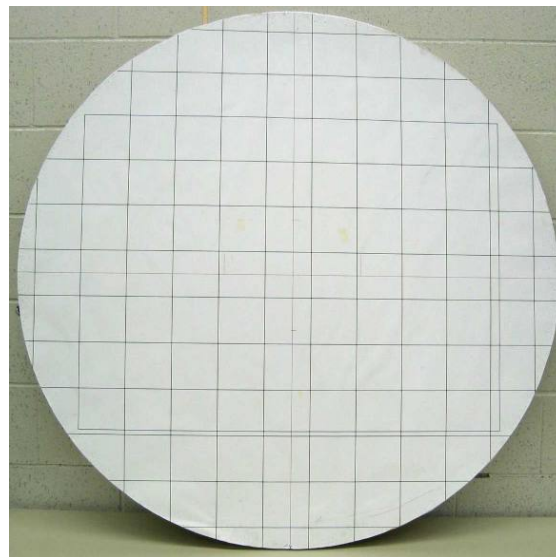


Figure 4.9 Calibration grid pattern.

Images of the grid pattern were recorded with the platform positioned at  $0^\circ$ . The images were enlarged using the image processing software (Matrox Inspector 2.0, Matrox Electronics Ltd, Dorval, QC) until the pixels could be distinguished. Different line markers of 450 to 600 pixels in length were superimposed on the vertical and horizontal lines forming the grid. If the starting and ending points of a vertical line marker weren't included in the same column (for vertical markers) or row (for horizontal markers) of pixels, the camera was slightly rotated. At the end of the process, the lines forming the grid pattern were parallel to the sides of the image when the platform was at  $0^\circ$ . When a pattern was installed on the rotating platform, four marks at 0, 90, 180 and  $270^\circ$  on the

periphery of the patterns were aligned with the corresponding angles on the central cross of the grid pattern (Fig. 4.10). This ensured not only that the pattern was centered on the platform, but also that the orientation of the rows was consistent with the grid calibration.

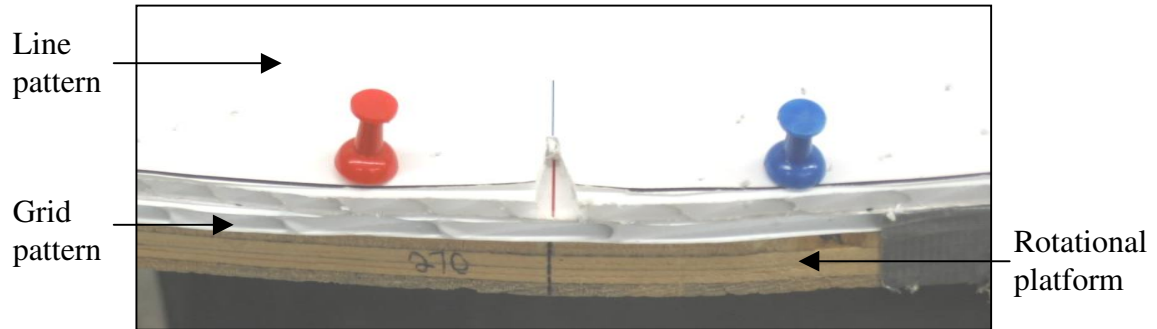


Figure 4.10 Line pattern aligned with the grid pattern on the rotational platform.

To be able to quantify the error in the determination of the location of the inter-row spaces at different angles, the centre of rotation of the patterns was centered in the image. To position the camera, a black sheet of paper was fixed on the rotary platform, with its centre exactly over the centre of rotation of the platform. The image processing software (Matrox Inspector 2.0, Matrox Electronics Ltd, Dorval, QC) analyzed images of the sheet and determined its centre by performing a blob analysis. If the centre of the sheet in the image wasn't at (319.5, 239.5), the camera was slightly moved until the centres matched. To confirm the right location, the platform was rotated at different angles and other images were gathered and analyzed.

A bubble level was used to ensure that the camera was sitting vertically over the platform, and that the XYZ $\theta$ -table and platform were level.

#### 4.1.4 Testing of the Line-detection algorithms

Testing of the Line-detection algorithms was done by subjecting them to images of parallel line patterns, with a specific row spacing (RS), row width (RW), offset and angle. Prior to acquiring each image, a pattern was chosen based on randomly assigned



values of row width (RW), row spacing (RS) and offset, fixed to the rotating platform, and rotated to a randomly assigned angle. Two images were recorded: the original non-weedy image and a duplicate of that image for safety. Simulated weeds were then disposed on the pattern manually, at random. Again, two images were recorded: one original, one duplicate. Each combination of RS, RW, offset and angle was repeated 3 times. A total of 288 original images were gathered under “weed free” conditions (2 RW x 2 RS x 4 offsets x 6 angles x 3 replicates), and 288 under “weedy” conditions. The complete database of images included 576 original images, and 576 duplicate images. The values describing each image gathered are presented in Table 4.1.

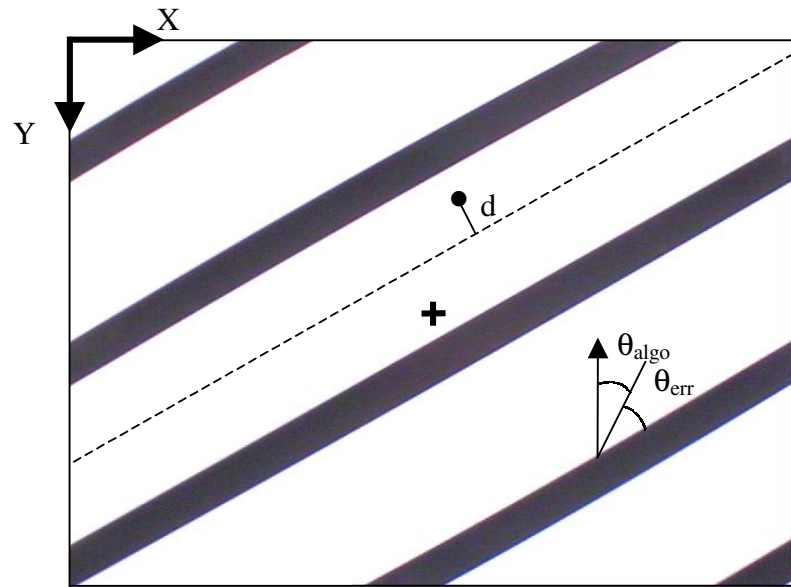
Table 4.1 Factors describing the images collected in Line-detection algorithms testing.

<b>Factor</b>	<b>Levels of factor</b>
Row width (RW)	51 or 76 mm
Row spacing (RS)	152 or 229 mm
Offset	0, 1/4RS, 1/2RS, 3/4RS (mm)
Angle	0, 30, 60, 90, 120 or 150°
Replicate	a, b or c
Noise	Present or absent
Image category	Original or Duplicate

The original images were processed by each Line-detection algorithm. When the file of an original image was damaged, the duplicate image was processed instead. This was the case for the original noise-free image described by a RW of 76 mm, a RS of 152 mm, an angle of 90°, an offset of 1/2RS, replicate b, and for the 2 original noisy images described by a RW of 76 mm, a RS of 152 mm and 229 mm, an angle of 60°, an offset of 1/4RS, replicate a and c, respectively. It was impossible to see the image contained in these “corrupted” files because they generated an error when opened by the Line-detection program.

The error in the algorithm results were quantified in terms of error in the angle of rows found, and positional error. The error in row angle was calculated as the absolute value of the difference between the angle found by the algorithm and the angle of the line

pattern. The positional error was defined as the perpendicular distance from the point (found by the algorithm) describing the inter-row space centreline to the closest centreline of a white stripe (Fig. 4.11). If the distance was 0, the target position, according to the algorithm, was directly over the centreline of a white stripe. In order to determine the positional error, the equations describing the centrelines of the white stripes were first determined using homogenous coordinates for each set of 3 replicates, from the values of RW, RS, offset and angle. Appendix C reports the calculations performed to find the equations of the inter-row space centrelines. The time required for the computer to complete the analyses using each of the algorithms was also recorded.



- + = image centre
- = location of the inter-row space centreline according to the algorithm
- d = positional error
- $\theta_{\text{algo}}$  = angle of rows determined by the algorithm
- $\theta_{\text{err}}$  = error on row angle

Figure 4.11 Representation of the row orientation error and positional error.

## 4.2 XYZ $\theta$ -table

The second objective of the present project was fulfilled by building an XYZ $\theta$ -table. This section describes the XYZ $\theta$ -table and its controls (Fig. D.1). The procedure followed to evaluate the XYZ $\theta$ -table is also presented.

### 4.2.1 Hardware and software

The XYZ $\theta$ -table (Fig. 4.12) consisted of a 2 m by 2 m square frame of steel supported by four legs. The legs were telescopic so that they could be adjusted to any length ranging from 0.75 m to 1.70 m. Images and specifications of the parts composing the table are given in Appendix D. Two rails in the X direction (X-rails) supported a rail in the Y direction (Y-rail), which supported a rack in the Z direction (Z-rail), which supported a rotational table ( $\theta$ -table). To manipulate the table, stepper motors were chosen because of their ease of control.

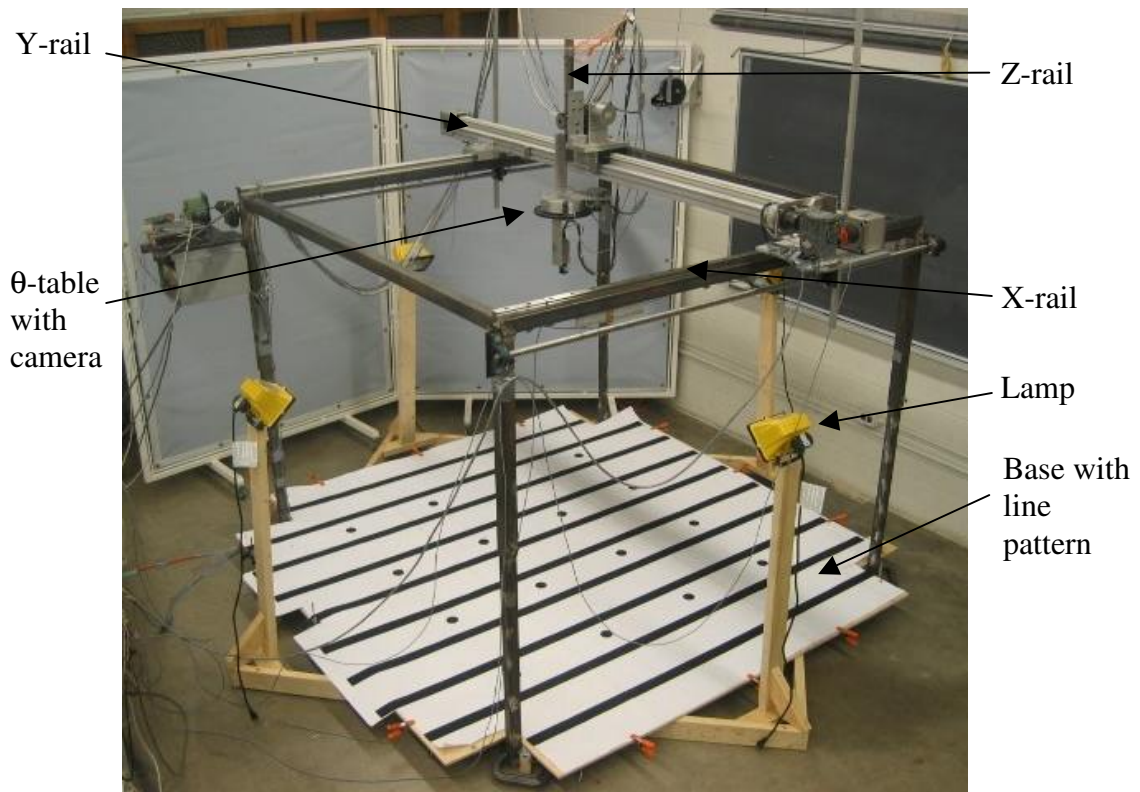


Figure 4.12 XYZ $\theta$ -table testing set.

Both X-rails were composed of 2 bearings and a 2-m rail, or guideway. One of the X-rails was composed of a carriage rolling on track roller linear bearings (#LFKL52 SF, INA Bearing Company Ltd, Oakville, ON) and an aluminium guideway (LFS52 C, INA Bearing Company Ltd, Oakville, ON). The other X-rail was composed of a carriage

rolling on recirculating linear glide bearings (#KWE30 G4V0, INA Bearing Company Ltd, Oakville, ON) and a steel guideway (#TKD30 G4, INA Bearing Company Ltd, Oakville, ON). The rails were fixed on the frame with a series of bolts. When fixing the rails to the frame, it was important to ensure that the X-rails were parallel. The bearing system on each rail was driven on the rails by the rotation of two 2.1-m ball screws (#KGS2550-050, INA Bearing Company Ltd, Oakville, ON) fixed at their ends by self-adjusting bearings bolted to the frame (#ZKLF2068 2RS, INA Bearing Company Ltd, Oakville, Ontario, and #GRAE20RRB, The Torrington Company, Torrington, Connecticut). A 23 VDC 0.66 A stepper motor (#21-4233D-23992, Sigma Instruments, Braintree, MA), presented in Figure D.2, was installed at one end of one of the ball screws. The motor was reduced by a ratio of 2.25:1 by a belt and pulley system (Fig. D.2). The other ball screw was driven by the first ball screw with a timing belt. This ensured that the Y-rail moved perpendicularly to the X-rails.

The Y-rail, fixed on the carriages of the X-rails, was a 2.68-m aluminium belt-driven rail (Linear actuator with track roller guidance system and toothed belt drive #MLF52155 ZRAR/2661-2170, INA Bearing Company Ltd, Oakville, ON). A carriage was attached to the belt and slid as the belt was moved under the power of the motor. The 4.5 VDC 1.4 A stepper motor (GPF 3945-2A, Fuji Electric Co., Tokyo, Japan), presented in Figure D.3, coupled to the shaft was reduced by a 60:1 right angle worm gear reducer (#361044, Renold Canada Ltd, Brantford, ON).

The Z-rail (Fig. D.4) was installed on the carriage of the Y-Rail. It consisted of a 508-mm rack (#TKV25ZHP, INA Bearing Company Ltd, Oakville, ON) driven by a 51-mm helical pinion gear (#24-22-330, INA Bearing Company Ltd, Oakville, ON) and sliding on 2 recirculating linear glide bearing carriages (#KWVE25HG4VI and #KWVE25SBG3V1, INA Bearing Company Ltd, Oakville, ON). A 4.5 VDC 1.4 A stepper motor (GPF 3945-2A, Fuji Electric Co., Tokyo, Japan), reduced by a 50:1 right angle worm gear reducer (#68356, J. Holroyd & Co. Ltd, Milnrow, England), was coupled to the pinion. The Z-rail could raise or lower the camera within a range of 0.33 m.

The  $\theta$ -table (Fig. 4.13) was attached at the lower end of the Z-rail (Fig. D.4). It consisted of an aluminium casing in which a 200-mm diameter thrust bearing (#CSXD 055, INA Bearing Company Ltd, Oakville, ON) was supporting a cylindrical aluminium free-rotating core. A 254-mm diameter gear was bolted on the external face of the core. A 4.5 VDC 1.4 A stepper motor (GPF 3945-2A, Fuji Electric Co., Tokyo, Japan) with a 38-mm pinion coupled to its shaft was driving the gear. The resulting reduction was 6.66:1. A vertical mount for the camera was extruded from a 101-mm diameter aluminium cylinder (Fig. 4.13). The camera mount was fixed to the centre of the gear. The camera was then bolted to the vertical mount. The  $\theta$ -table was designed to have the rotational axes of the aluminium core, the gear and the camera mount aligned with the centre of the lens of the camera.

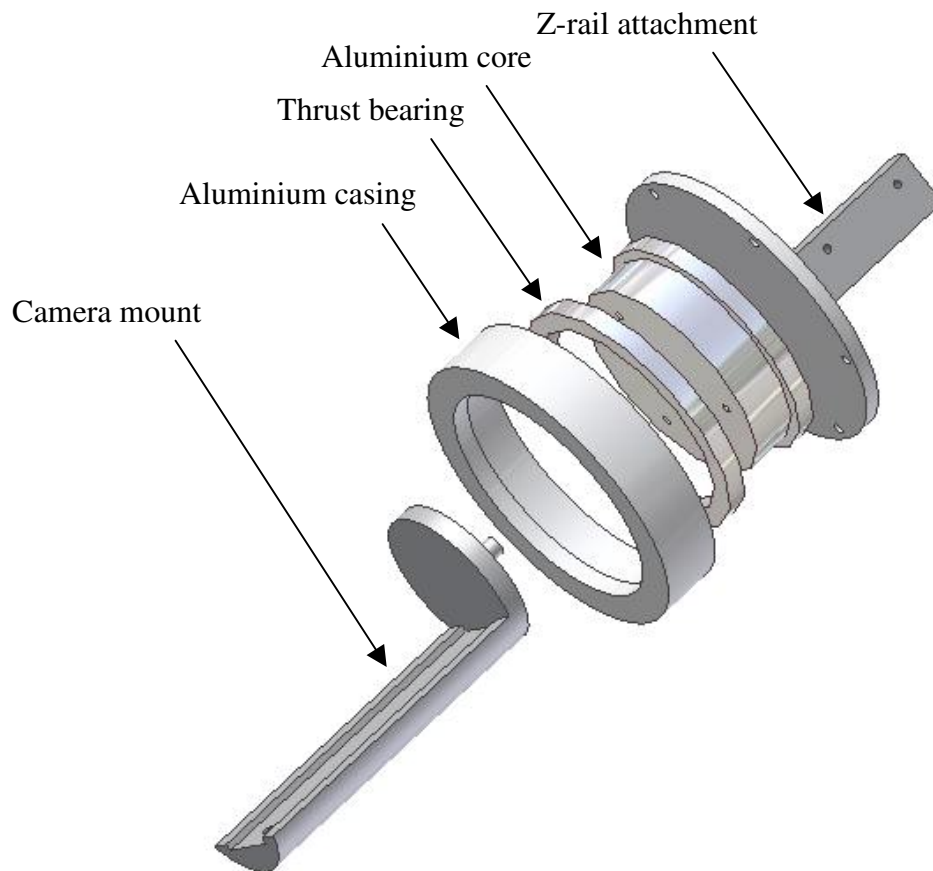


Figure 4.13  $\theta$ -table.

The vision hardware (camera, framegrabber) and software (imaging library) used to perform the tests was the same as the one described in Section 4.1.1. A 2.45 m x 2.45 m plywood base was built and installed under the table. That base was covered by a line pattern (Fig. 4.14) of the same dimensions, plotted as three 0.82 m x 2.45 m separate drawings later assembled. The line pattern represented crop rows of 229 mm in row spacing and 51 mm in row width. 16 “target positions” were equally distributed on the pattern (Fig. 4.14). A target was a black dot of 51 mm in diameter. The lines on the sheets were used as a reference to evaluate the camera rotation, and the targets were used as references to evaluate the positioning of the camera.

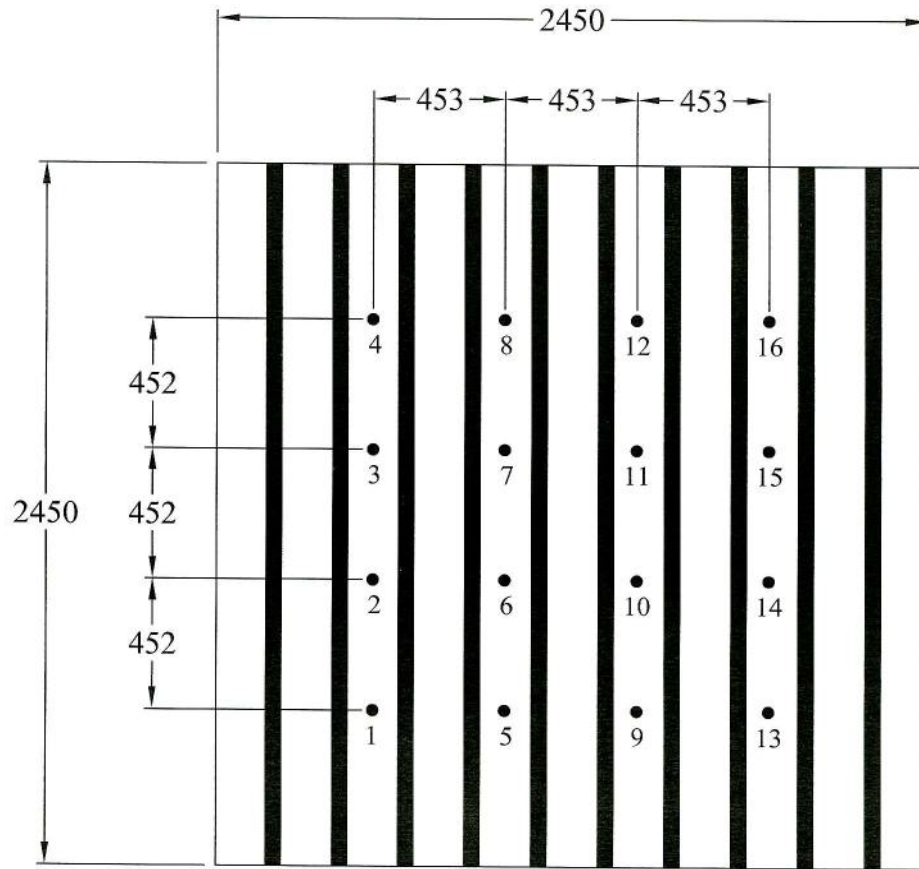


Figure 4.14 Line pattern used in evaluation of the XYZ $\theta$ -table (dimensions in mm).

The camera was positioned at a height of 1.22 m over the platform. This resulted in an image size of 0.92 x 0.69 m, thus a resolution of 10 mm per 7 pixels. Uniform lighting

was provided by four 500-W halogen lamps (Weatherproof Portable Halogen Work Light #60504, Globe Electric Company Inc., Pointe-Claire, QC) fixed on each side of the platform (Fig. 4.12).

Each motor was controlled by a commercial controller board (Stepper Motor Driver Module, Weeder Technologies, Ft Walton Beach, FL), presented in Figure 4.15. All boards were daisy-chained together, connected to the RS-232 serial port of a desktop computer (Pentium III 833 MHz), and powered by a 30 VDC power supply in the case of the Motor X, and 24 VDC power supplies for the other motors. Microsoft Visual Basic 6.0 was used to communicate with the computer serial port. The boards were all set to a different address and communicated independently with the host computer via different alphanumeric command strings. The user could actuate the motors to a specific position within the range 0 to 16 777 215 steps, change the speed, the acceleration and current position of the motor. Figure 4.16 presents a schematic of the controls of one motor and 2 switches. Additional images and schematics of the controls are provided in Appendix D.2.

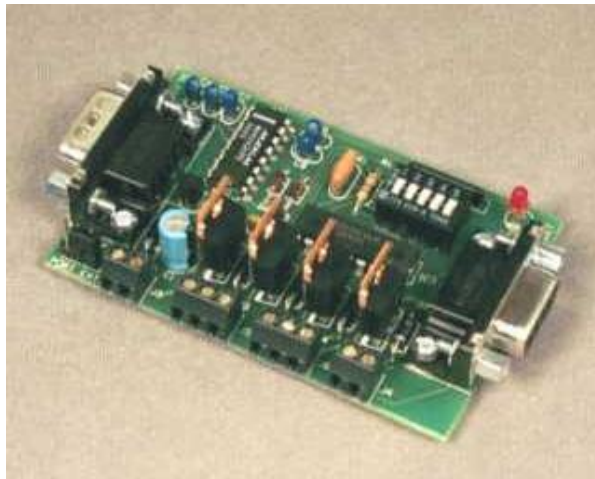


Figure 4.15 Stepper motor controller board (Weeder Technologies, 2004).



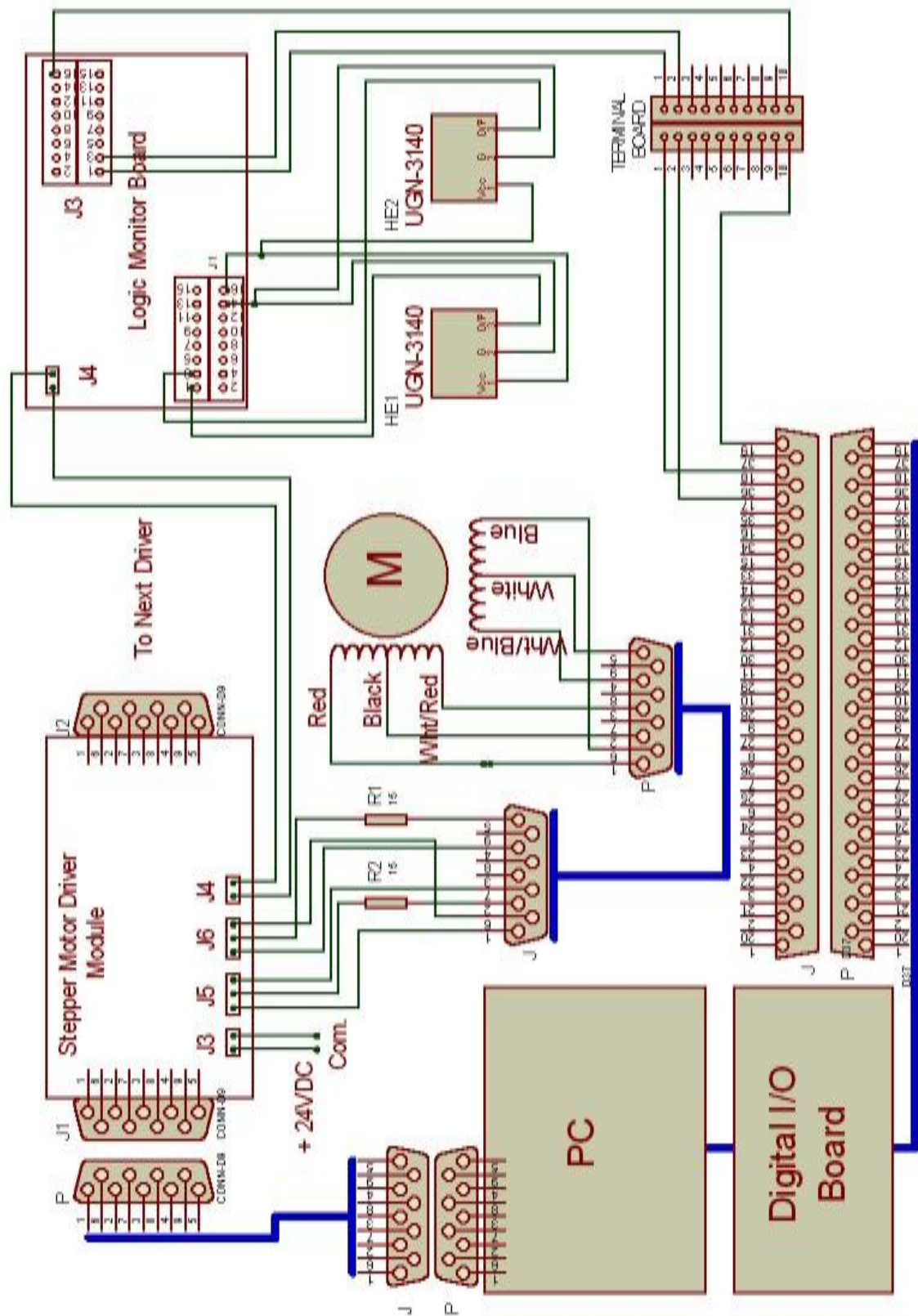


Figure 4.16 Simplified block diagram of the control system of the XYZθ-table.



The controller boards were built in a way that allowed them to sense the advance of a motor by one or more normally-open limit switches connected in parallel to the board (Fig. D.7). A Hall-effect limit switch (UGN3140U228, Allegro MicroSystems Inc., Worcester, MA), presented in Figure 4.17, was installed at each end of one X-rail and the Y-rail. Two switches were installed diametrically opposite to each other on the  $\theta$ -table (Fig. D.15). On the X-rail, Y-rail and  $\theta$ -table, one switch was used as a ‘home’ position to reset the motor to a known position, while the other was used as an emergency stop to prevent the Z-rail from hitting the frame and to avoid the camera from rotating infinitely.

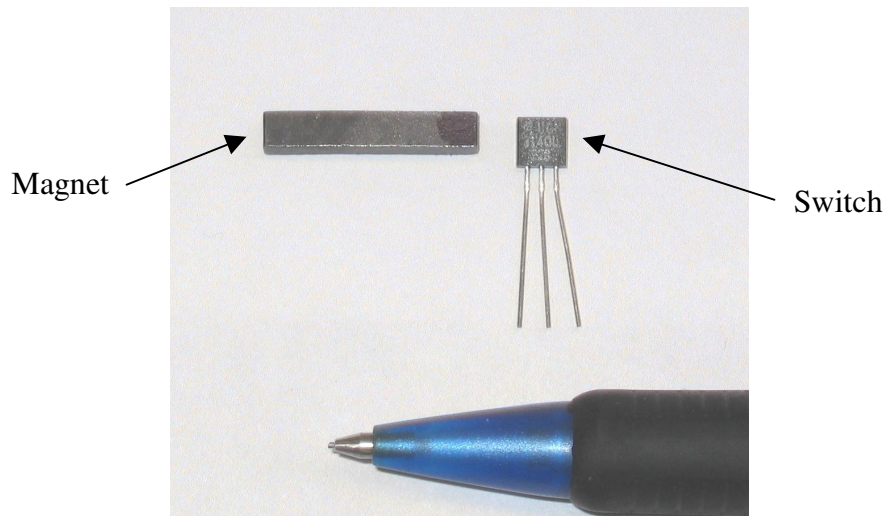


Figure 4.17 Hall-effect switch and magnet.

To be able to read the status of these switches, a digital I/O PCI board (PCI-DIO24, Measurement Computing, Middleboro, MA) was installed on the computer. This board had 24 bidirectional lines, i.e. the user could read and write to each line. Complete specifications and schematics are provided in Appendix D.2. Three logical circuit boards were built allowing connecting, reading and disabling each up to 8 Hall-effect switches for the motors driving the X, Y and  $\theta$  axes. These circuits improved the versatility of the controls. For example, it might become useful later in the project to connect more than 2 limit switches on the same rail or  $\theta$ -table. The additional switches, equally spaced on the rail or  $\theta$ -table, could be used to confirm the position of the motor, in which case they

may need to be disabled when returning to the ‘home’ position. They could also mark different ‘home’ positions, therefore reducing the displacement required to reset a motor. The schematic of the logical circuit and a picture of one circuit are provided in Appendix D.2.

Commercially available software (Universal Library 32 bit version 1.0, ComputerBoards Inc., Middleboro, MA) was installed on the computer to allow writing a personalized program acquiring data from the digital I/O PCI board. The functions included in the library were compatible with Microsoft Visual Basic 6.0.

Finally, to facilitate connecting and disconnecting switches to the digital I/O PCI board, a screw terminal board was built (Fig. D.6). This board had the same pinout as the digital I/O PCI board (Fig. D.12).

To control the XYZ $\theta$ -table, a program was written in Visual Basic (Appendix D.3). The Controller program was similar to the Line-detection program in the way that it incorporated the same Image Collecting block and a Processing block including the Stripe Analysis algorithm. The Blob Analysis algorithm was also included in the Processing block to double-check results of the Stripe Analysis algorithm, if necessary. A new algorithm, the Target algorithm, was added to the Processing block. This algorithm, similar to the Blob Analysis algorithm, used the blob analysis commands of the imaging library (Matrox Imaging Library version 4.0, Matrox Electronics Ltd, Dorval, QC) in order to find the centre of the target closest to the centre of the image. The Controller program also featured an Actual Parameters window and a Results window similar to the ones included in the Line-detection program (Fig. 4.3). Three new windows were created: the Data Acquisition window, the Manual Control window and the XYZ $\theta$ -table Control window. The Data Acquisition window allowed reading the status of each Hall-effect switch simultaneously and the Manual Control window was used to actuate the motors independently, change and verify their positions, speed and acceleration.

The most important window was the XYZ $\theta$ -table Control window. The code of the Control window incorporated the code of the Data Acquisition and Manual Control windows to create a more comprehensive interface to perform the tests. The Control window featured a set of 16 option buttons representing the targets on the line pattern. Six other option buttons represented the different rotation angles around the  $\theta$ -Table. For each image to be collected, the user could choose the location and angle of rotation of the camera. Three command buttons allowed sending the camera back to its ‘home’ position on the X-rails, Y-rail and  $\theta$ -table, and three other reset the motors to a specific step position once they were at the home position. This feature helped to prevent positioning errors due to possible slippage of the motors. Finally, 6 checkboxes represented the Hall-effect switches. Their status (on/off) could be inquired at all times. Figure 4.18 presents the XYZ $\theta$ -table Control window of the Controller program.

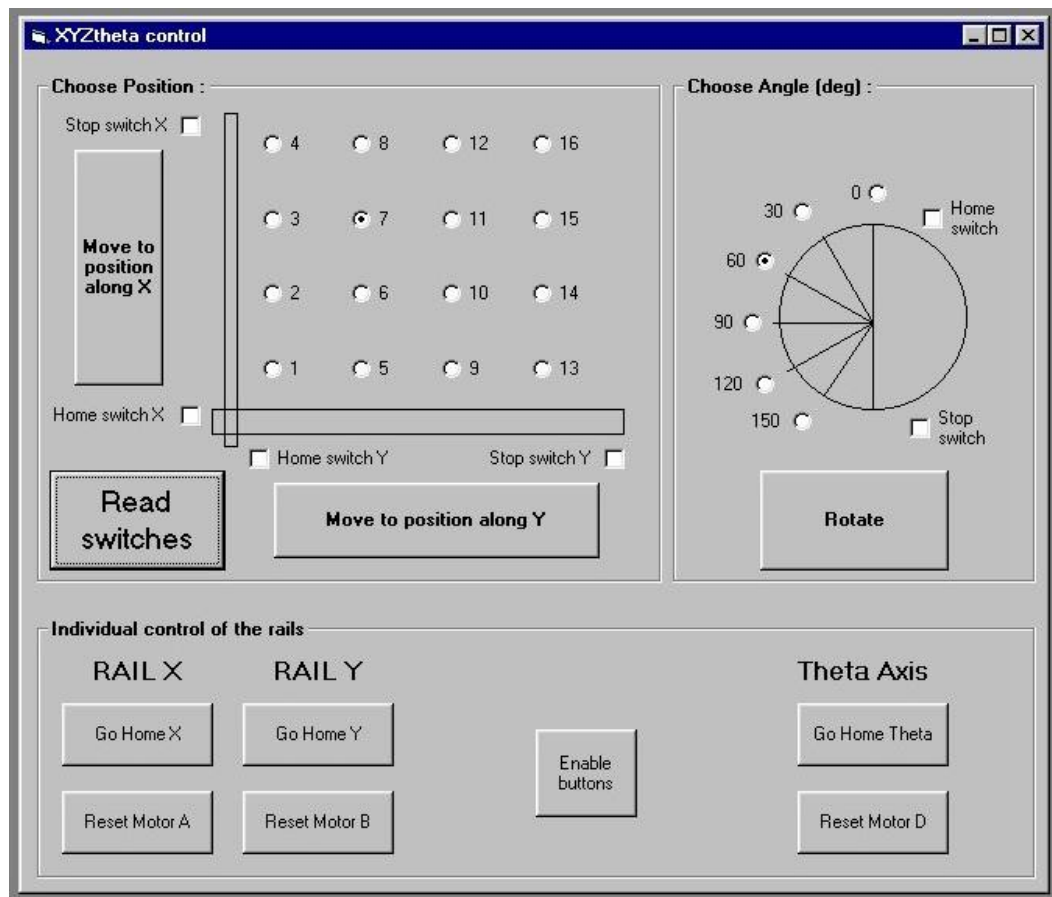


Figure 4.18 XYZ $\theta$ -table Control window of the Controller program.

#### 4.2.2 Calibration of the testing set

When installing the line pattern on the platform, proper care was given to the orientation of the pattern relative to the XYZ $\theta$ -table. It was desired that the stripes be parallel to the X-rail closest to the ‘home’ position on rail Y ( Fig. 4.19).

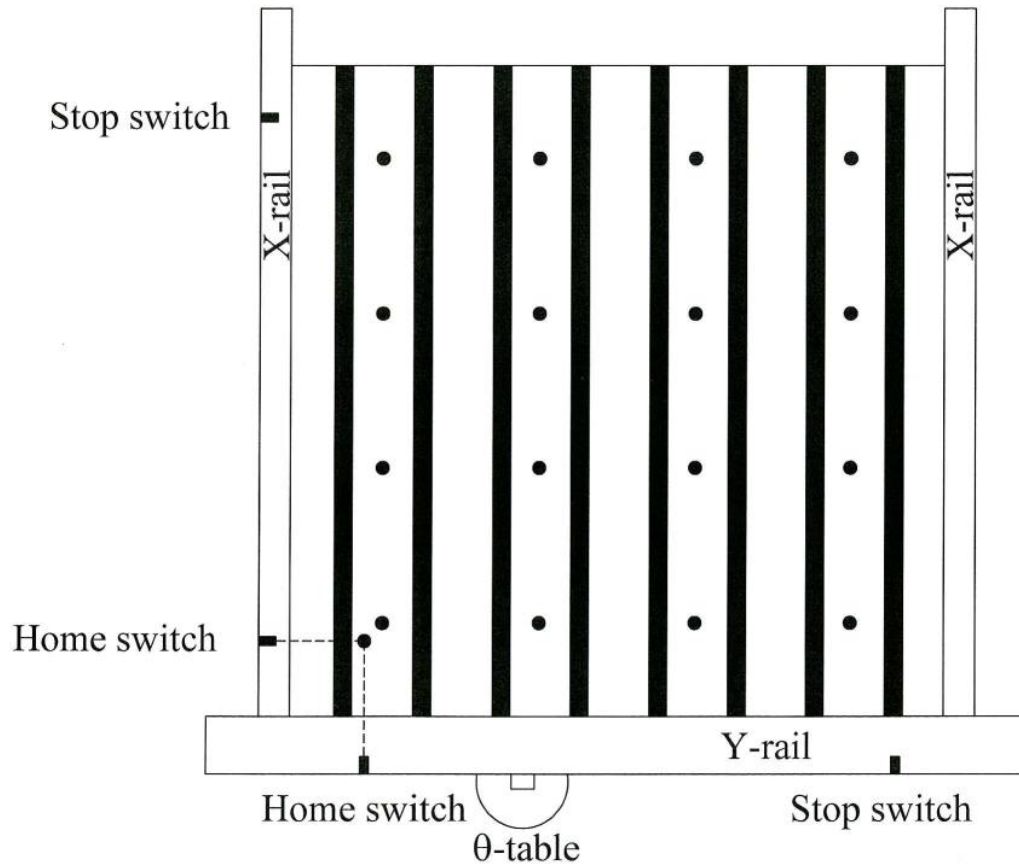


Figure 4.19 Home positions on rails X and Y.

To achieve this goal, the camera was positioned near the X-rail of interest, at an arbitrary angle. An image of the pattern was recorded at both ends of the X-rail without moving the camera on the Y-rail and  $\theta$ -table. Both images were analyzed in the following way. A black stripe in the first image was chosen to be the reference. Using commercial image processing software (Matrox Inspector 2.0, Matrox Electronics Ltd, Dorval, QC), the images were enlarged until the pixels were apparent. The pixel columns

corresponding to the edges of the reference stripe were recorded for the first row of pixels (top of the stripe) and the 479<sup>th</sup> row of pixels (bottom of the stripe). If the pixel columns were not consistent in both images, the pattern was slightly rotated and another series of images was recorded. This procedure was repeated 5 times. At the fifth attempt, the line pattern was considered parallel to the X-Rail because the edges of the reference stripe were consistently positioned in the images collected at both ends of the X-rail. Five other pictures were gathered at different locations on the X-rail to confirm the calibration. The pattern was then fixed to the platform using clamps.

The position increments of the motors moving the camera along the X and Y rails in terms of mm/step was determined. The motor was moved along each rail of a large number of steps (5000 to 12000 for the X-rail, 20000 to 56000 for the Y-rail), and the distance travelled was measured. Measurements were repeated 5 times for each rail. In the X direction, the mean distance travelled was 0.097 mm/step, and in the Y direction, it was 0.025 mm/step. To determine the positioning resolution of the  $\theta$ -table in terms of degrees/step, the camera was first rotated to have the stripes appearing vertically in the image. To confirm that the stripes were vertical, the image was enlarged using the image processing software (Matrox Inspector 2.0, Matrox Electronics Ltd, Dorval, QC), and the pixel columns corresponding to the edges of the black stripe were recorded for the top and the bottom of the stripe. If the pixel column was the same at the top and bottom of the stripes, the stripes were considered to be at 0° in the image. The camera was then rotated, using the motor on the  $\theta$ -table, until the stripes were at 90°, 180°, 270° and 360°. Once again, the angles were confirmed by looking at the columns or rows of pixels corresponding to the stripe edges. When an angle was confirmed, the number of steps rotated from 0° was recorded. All images provided a positioning resolution of 0.3°/step.

The location of the home positions relative to the targets was determined. When the camera was at the home position on the X-rail, Y-rail and  $\theta$ -table, target #1 was not at the centre of the image, or (319.5, 239.5) in pixel coordinates. The point (319.5, 239.5) in the image was considered to represent the location of the camera. To retrieve the location of the camera on the reference line pattern, another target, printed on a separate

sheet of paper, was added on the pattern. This "home target" was centered in the image by trial and error, with the camera at the home position on the X-rail, Y-rail and  $\theta$ -table. The centre of the target was determined manually by enlarging the image, binarizing it, and observing the position of the black pixels horizontally and vertically. When the target was centered in the image, it was fixed to the pattern, and the camera was moved along the X and Y rails and back to the home position to confirm the results. The distance between the centre of the home target and target #1 was measured. The home target was at 46.5 mm from target #1 along the X-rails, and 48.5 mm from target #1 along the Y-rail. From the knowledge of the locations of all targets relative to target #1 (Fig. 4.14), it was possible to know the distance between each target and the home position. These distances, converted to the number of stepper motor steps, were used to describe each target in the Control block of the Controller program.

The location of the home position on the  $\theta$ -table was determined based on two conditions. First, it was desired, for arbitrary reasons, that the rotation from the home position on the  $\theta$ -table to all rotation angles be in the same direction. Because the Stripe Analysis program provided the angle of the rows from a vertical axis in the image ( $0^\circ$ ), with the angles increasing clockwise, the direction of rotation of the camera had to be counterclockwise. Second, to be able to evaluate the rotation of the camera at 6 different angles ( $0, 30, 60, 90, 120$ , and  $150^\circ$ ), it was desired that none of the angles correspond to the home position on the  $\theta$ -table. The methodology used to determine the home position satisfying both conditions was the following. The camera was first rotated to have the stripes approximately vertical in the image. Then the camera was rotated clockwise an arbitrary number of motor steps. The home switch was fixed at the resulting location on the  $\theta$ -table. A stop switch was also fixed on the  $\theta$ -table, at approximately  $180^\circ$  from the home switch.

It was also necessary to determine the distance in steps of the motor from the home position to each of the rotation angles. To accomplish that, the angle of the stripes with the camera at the home position was determined using the Stripe Line-detection algorithm. Seven images were recorded at arbitrary locations over the linear pattern,

with the camera at the home position. It was determined that, on average, the stripes were at  $-13.6^\circ$  from the vertical, with the angles increasing clockwise. The positioning resolution of the  $\theta$ -table ( $0.3^\circ/\text{step}$ ) allowed calculating the number of steps required to have the stripes at 0, 30, 60, 90, 120, and  $150^\circ$  in the image. These values were entered in the Controller program and used to control the motor rotating the  $\theta$ -table.

The calibration of the camera height was accomplished using the same method described in Section 4.1.3. Again, the camera was sitting at 1.22 m from the platform. This resulted in an image size of 0.92 x 0.69 m and a resolution of 10 mm per 7 pixels.

While calibrating, it was observed that the camera might not be rotating around the centre of the image. At the time, the reason was believed to be a slight tilt of the camera. Some shimming was inserted between the  $\theta$ -table and the 254-mm gear in an attempt to eliminate that tilt, which seemed to improve the rotation.

#### **4.2.3 Testing of the XYZ $\theta$ -table**

Testing of the accuracy of the XYZ $\theta$ -table was done by moving the camera in the XY-space to a randomly assigned location, or target, and rotating it to a randomly assigned orientation. Six rotation angles were tested: 0, 30, 60, 90, 120, and  $150^\circ$ . Two images were gathered at each location: an original and a duplicate. The camera was then sent back to the home position on the X and Y rails, and on the  $\theta$ -table, and the motors' positions were reset. With 3 replicates per combination of target location and angle, a total of 288 original images were gathered (16 targets X 6 angles X 3 replicates), and 288 duplicate images. Table 4.2 presents the factors describing the images gathered.

Table 4.2 Factors describing the images collected in XYZ $\theta$ -table testing.

<b>Factor</b>	<b>Levels of factor</b>
Target location	#1 to 16
Rotation angle	0, 30, 60, 90, 120 or 150°
Replicate	a, b or c
Image category	Original or Duplicate

The accuracy in the displacement and rotation was measured from the images gathered. The original images were processed by the Stripe Analysis algorithm to determine the angle of the stripes in the image, and by the Target algorithm to determine the location of the target closest to the centre.

The error in camera rotation was quantified in terms of error in the angle of rows found. The error in row angle was calculated as the absolute value of the difference between the angle found by the Stripe Analysis algorithm and the assigned orientation of the camera.

The error in camera positioning was quantified in terms of distance between the centre of the image to the centre of the target of interest. That distance was calculated as the hypotenuse of the error in the X direction and error in the Y direction in the image. The coordinates of the target centre were provided by the Target algorithm, whereas the image centre was considered to be at (319.5, 239.5), with the origin in the upper left corner of the image (Fig. 4.20). The error distance, in pixels, was converted to mm with the resolution of the system (10 mm: 7 pixels). An error in positioning equal to 0 meant that the camera was perfectly positioned over the target.



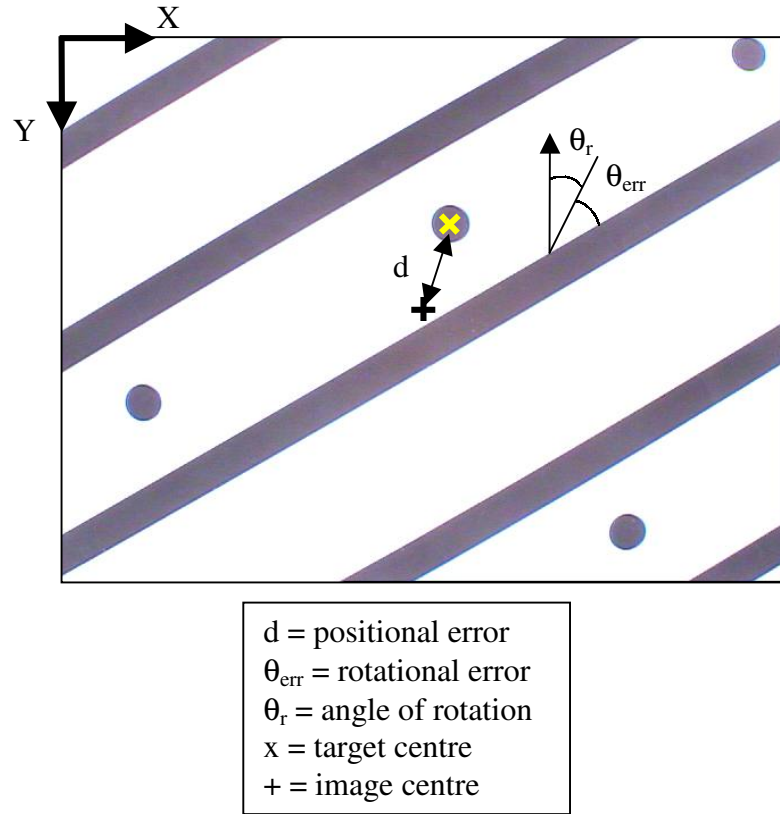


Figure 4.20 Representation of the positional error and rotational error in images gathered for XYZ $\theta$ -table testing.

To help visualize the camera position at all target locations during the tests, an orientation was attributed to each positioning error (Appendix E.1.2). This involved the determination of the error along the X-rails and Y-rail, in a  $X^B Y^B$  coordinate system where axis  $X^B$  was parallel to the X-rails and the stripes. However, because of the rotation of the camera, the  $X^A Y^A$  image coordinate system was not the same as the  $X^B Y^B$  line pattern coordinate system (Fig. 4.21). To be able to describe the two points of interest (the target centre and the image centre) in the  $X^B Y^B$  coordinate system, a rotation matrix was applied on the coordinate system  $X^A Y^A$ . From the knowledge of the actual coordinates of the targets in the  $X^B Y^B$  line pattern coordinate system, it was then possible to calculate the components of the positioning error along  $X^B$  and  $Y^B$  (or along the X and Y rails). From these values, the direction of the camera relative to the target was calculated. The direction was expressed as 0 to 180° clockwise from the X-rails and

0 to  $-180^\circ$  counterclockwise.

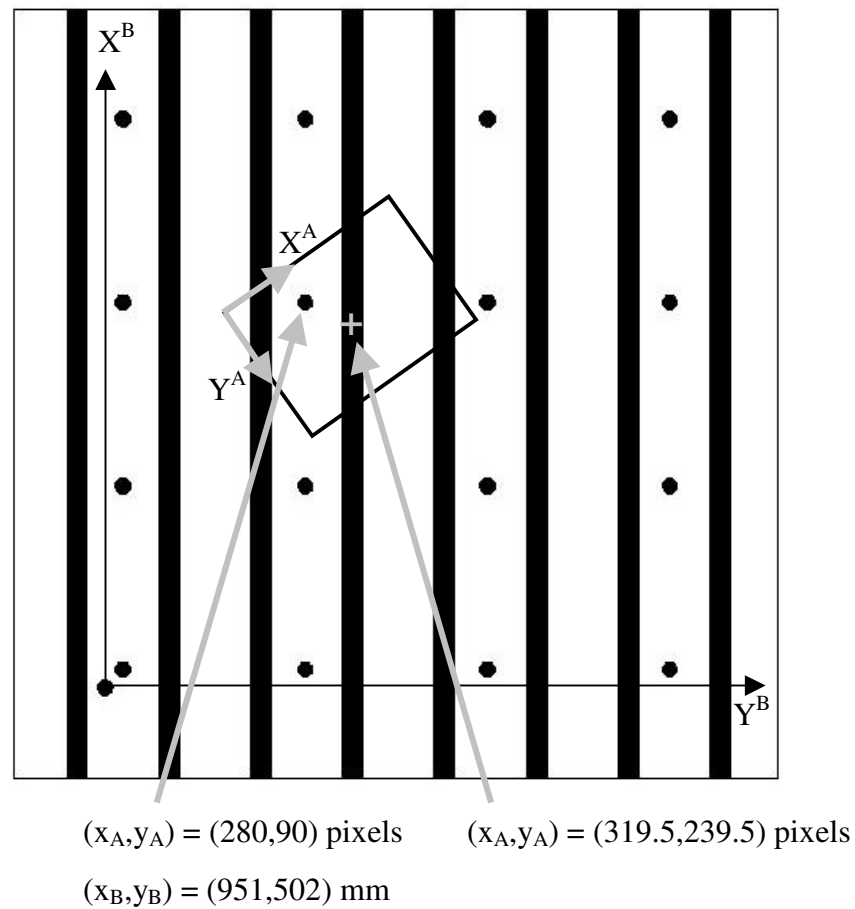


Figure 4.21 Representation of the coordinate systems of the XYZ $\theta$ -table and the images.

It was observed that for the 3 replicates described by a common target position and camera rotation angle, the camera was always approximately in the same direction. That allowed calculating a mean ‘direction angle’ of the camera for the 3 images (Fig. 4.22). From the mean direction and the previously calculated mean positioning error, it was possible to calculate the mean location of the camera in the  $X^B Y^B$  line pattern coordinate system for each set of 3 replicates (Appendix E.1.2). The physical camera location points helped to determine if the error was due to the X-rails or the Y-rail, and to understand the behaviour of the camera during the rotation.

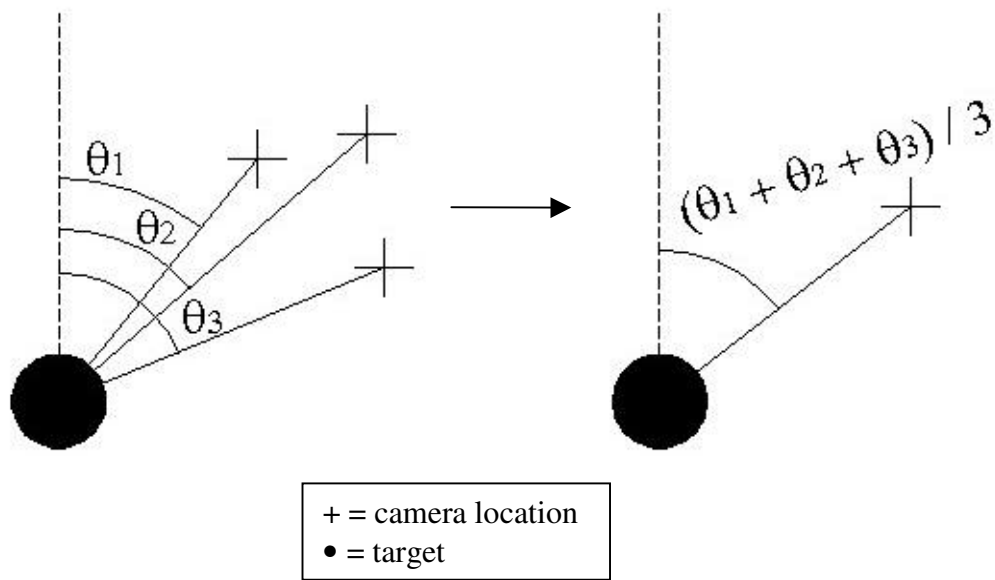


Figure 4.22 Calculation of the mean direction angle of the camera relative to a target.

## 5. RESULTS AND DISCUSSION

This section presents the data and results of the evaluation of the Line-detection algorithms and the XYZ $\theta$ -table.

### 5.1 Line-detection algorithms

Figure 5.1 presents examples of the images gathered to evaluate the Line-detection algorithms. The original raw and processed data are presented in Appendix E.1.1.

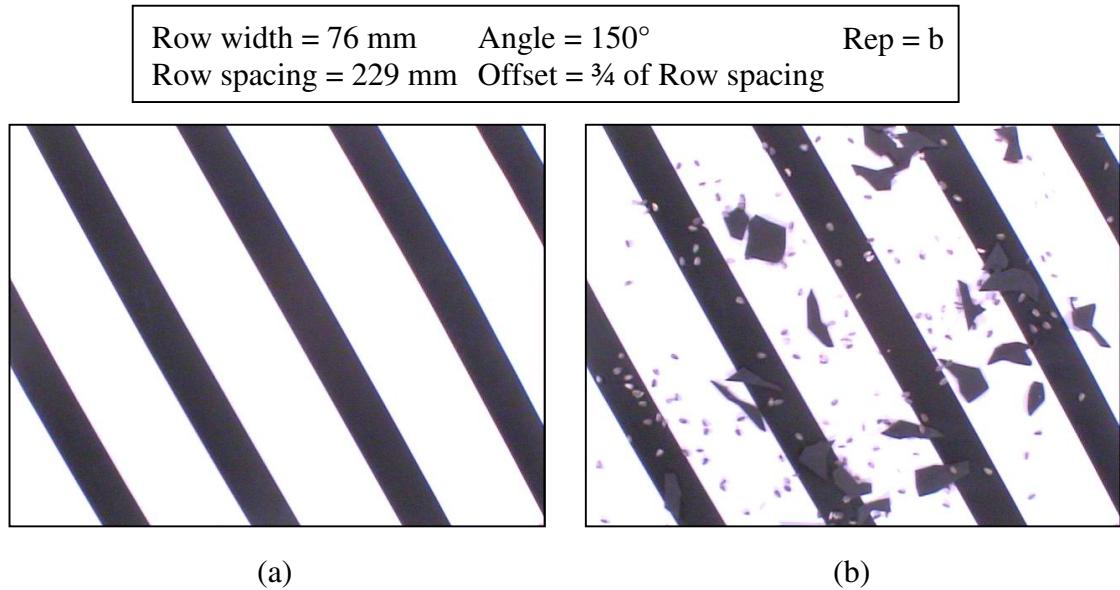


Figure 5.1 Example of images collected to evaluate the Line-detection algorithms.  
Original image without noise (a), and with noise (b).

The Line-detection algorithms were evaluated on the basis of their ability to determine the angle of the stripes in the images gathered, their ability to locate the centreline of a white stripe (inter-row space), and their processing time.

### 5.1.1 Determination of stripe orientation

Mean errors for stripe angle determination were calculated for each algorithm, for noisy and noise-free images (Table 5.1). An analysis of variance (ANOVA) was performed on the noise-free and noisy data to investigate differences among the means of each algorithm. The statistical analysis software used for all ANOVA calculations was SAS v.8.02 (SAS Institute Inc., Cary, NC). The complete ANOVA programs and outputs are presented in Appendix E.2.1.

Table 5.1 Error in the determination of the row angle (deg) in original set of data,  $n = 288$ .

Algorithm	Noise-free images		Noisy images		
	Mean	Std dev	Mean	Increase	Std dev
Blob Analysis	0.47 <i>a</i>	0.36	0.96 <i>a</i>	104%	4.28
Hough Transform	0.49 <i>a</i>	0.43	0.85 <i>b</i>	73%	6.08
Stripe Analysis	0.54 <i>b</i>	0.36	0.54 <i>ab</i>	0%	0.41
Linear Regression	0.56 <i>b</i>	0.36	1.08 <i>c</i>	93%	1.13

Note: Means grouped according to the type of image (noise-free or noisy).

Grouping is based on a 95% LSD in a non-parametrical analysis.

Increase = ((mean noisy - mean noise-free) / mean noise-free) x 100

With noise-free images, the best algorithm was the Blob Analysis and it presented an error of  $0.47^\circ$ . Tests for normality rejected the null hypothesis that the data were normally distributed, both for the noise-free and noisy data set. Therefore, multiple comparisons of the means were performed in a non-parametrical analysis, which analyze the ranks of each datum rather than the original value (SAS Institute Inc., 1999). Whenever the results of the non-parametrical multiple comparisons were different than the parametrical analysis, the non-parametrical results were presented in the tables. In the case of Table 5.1, a 95% least significant difference (LSD) calculated in the non-parametrical analysis suggested that the mean error of the Blob Analysis algorithm was not significantly different from the mean error of the Hough Transform algorithm. The worst algorithm, using linear regression, had a mean error of  $0.56^\circ$ . The variation in the data was consistent for the four algorithms, with a standard deviation of approximately

0.4°.

Mean errors were larger when the noisy images were analyzed. The greatest increase in mean error (relative to the mean errors using the noise-free images) was posted using the Blob Analysis algorithm, with a 104% increase. The standard deviation of the mean also increased dramatically. The worst algorithm was again the Linear Regression algorithm, with a mean error of 1.08°. It is interesting to note that the Stripe algorithm mean wasn't affected at all by the noise in the images. The mean error of this algorithm remained unchanged, while the standard deviation increased slightly. In this case, the multiple comparisons of the non-parametrical analysis were presented, which explains the aberration of the means grouping (the mean 0.54 being grouped both with the mean 0.96 and the mean 0.85).

The general increase in data dispersion observed in noisy images raised some questions, especially concerning the Hough Transform algorithm. An analysis of the outliers was performed by calculating the standardized residuals (RStudent) (Appendix E.2.1). The RStudent statistic is defined as the residual divided by the root error mean square (Montgomery, 1991). When the RStudent value was greater than 3 in absolute value, the datum was considered an outlier. This is true when the residuals are normally distributed, which was not the case for these data. Therefore, one must be careful when using the results of the outlier analysis. It was suspected that the outliers in the data might be partly responsible for the increase in error mean and spread of the data. To confirm that hypothesis, the outliers in each sub dataset were removed and the ANOVA was performed again (Table 5.2). The same procedure was applied to the noise-free data. Without the outliers, the data were unbalanced. Therefore, least-squares means (lsmeans), which estimate the mean over a balanced data set (SAS Institute Inc., 1999) were calculated instead of arithmetic means. The outliers removed are tabulated in Table 5.3.

Table 5.2 Error in the determination of the row angle (deg), outliers removed.

Algorithm	Noise-free images			Noisy images			
	<i>n</i>	LSMean	Std dev	<i>n</i>	LSMean	Increase	Std dev
Blob Analysis	287	0.47 <i>a</i>	0.36	285	0.57 <i>a</i>	21%	0.44
Hough Transform	283	0.49 <i>a</i>	0.43	287	0.49 <i>b</i>	0%	0.42
Stripe Analysis	287	0.53 <i>b</i>	0.36	286	0.52 <i>ab</i>	-2%	0.38
Linear Regression	287	0.56 <i>b</i>	0.36	285	1.01 <i>c</i>	80%	0.87

Note: LSMeans grouped according to the type of image (noise-free or noisy).

Grouping is based on 95% LSD in a non-parametrical analysis.

Increase = ((mean noisy - mean noise-free) / mean noise-free) x 100

Table 5.3 Outliers removed from the data in row angle analysis.

Algorithm	Images	RW mm	RS mm	Angle deg	Offset [0,1,2,3]/4RS	Rep
Stripe Analysis	Noise-free	51	229	150	3	c
	Noisy	51	229	150	3	c
		51	229	30	0	a
Linear Regression	Noise-free	51	229	150	3	c
	Noisy	51	229	150	3	b
		51	229	150	1	c
		76	229	30	0	b
Hough Transform	Noise-free	51	152	90	3	b
		51	229	150	2	b
		51	229	150	3	c
		76	152	150	3	b
		76	229	0	2	a
	Noisy	76	229	30	0	b
Blob Analysis	Noise-free	51	229	150	3	c
	Noisy	76	152	30	3	a
		76	229	150	2	a
		76	229	90	1	c

Note: RW = Row width, RS = Row spacing.

The same outlier was removed from all noise-free sub data sets (RW = 51 mm, RS = 229 mm, Angle = 150°, Offset = 3/4RS, Rep = c). It is believed that this particular image was different than its two other replicates due to a slight error when installing the line pattern on the platform, thereby causing an error for all 4 algorithms. Additional outliers were removed in the noise-free data set of the Hough Transform algorithm. A visual

inspection of the noise-free outlier images didn't reveal any specific feature that might have caused the errors. Moreover, the errors associated with these outliers were very small. Therefore, it is believed that the outliers removed were again probably caused by a misalignment of the pattern on the rotational platform. Overall, the removal of the outliers didn't change significantly the noise-free results.

On the other hand, removing the major outliers in the noisy data greatly improved the results, especially in the case of the Hough Transform and Blob Analysis algorithms. In both cases, the mean error decreased by approximately 40%. In the case of the Hough Transform, the new mean for noisy images was the same as the mean for noise-free images. Moreover, the standard deviation of these algorithms was significantly reduced. This suggests that removal of the outliers greatly affected the results in the first analysis.

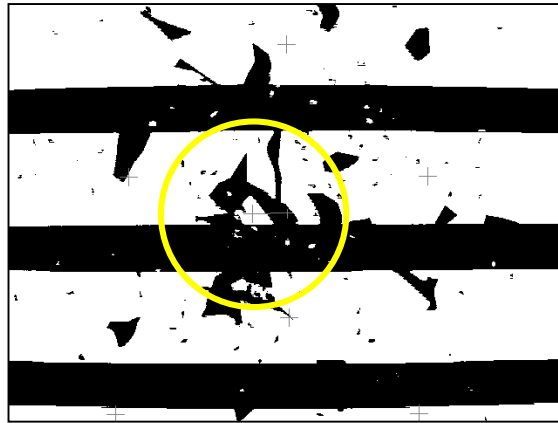
Figure 5.2 presents the processed noisy outlier images corresponding to the Blob Analysis algorithm. In those images, a line was drawn from the centre of the image to the closest blob centre. In the case of images in Figure 5.2 (a) and 5.2 (b), the dispersion of the weeds created a small white blob within an inter-row space. In both cases, because the centre of that blob was the closest to the centre of the image, it was selected to calculate the angle and position of the inter-row space. The image in Figure 5.2 (c) presents a case where the dispersion of the weeds created 2 'bridges' between consecutive rows. The bridges separated the inter-row space into 3 white blobs. The orientation of the longest axis of the blob closest to the image centre wasn't representative of the inter-row space orientation. However, because the blob and the inter-row space had the same width, the determination of the inter-row space centreline should be accurate (see Section 4.1.2).





(a)

Row width = 76 mm  
 Row spacing = 152 mm  
 Angle =  $30^\circ$   
 Offset =  $\frac{3}{4}$  of Row spacing  
 Rep = a  
 + = blob centre



(b)

Row width = 76 mm  
 Row width = 229 mm  
 Angle =  $90^\circ$   
 Offset =  $\frac{1}{4}$  of Row spacing  
 Rep = c  
 + = blob centre



(c)

Row width = 76 mm  
 Row spacing = 229 mm  
 Angle =  $150^\circ$   
 Offset =  $\frac{1}{2}$  of Row spacing  
 Rep = a  
 + = blob centre

Figure 5.2 Outlier images in the determination of stripe orientation by the Blob Analysis algorithm.

Figure 5.3 presents a processed outlier image common to the Hough Transform algorithm and the Linear Regression algorithm. The white crosses on the rows represent the centres of the rows found by edge detection. Three clusters of row centres are circled and identified. In this image, some “weed patches” were big enough and close enough to the crop rows to be classified as part of the rows. This caused the Similarity Matrix to form 3 groups instead of one to represent one of the rows. Cluster #2 wasn’t considered in the calculations of angle and positions of the rows because it was composed of only 7 centres. After defining the most popular line passing through all clusters, the algorithm determined that the lines formed by cluster #1 and #3 were the closest to the centre of the image. This caused a great error in orientation and positioning for the Hough Transform and Linear Regression algorithms.

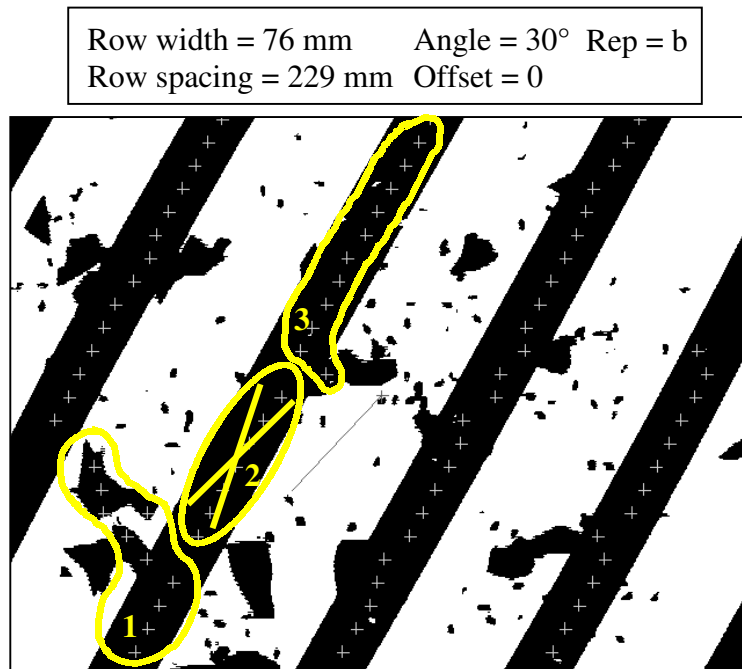


Figure 5.3 Outlier image in the determination of stripe orientation by the Hough Transform and Linear Regression algorithms.

The results of the Linear Regression didn’t improve significantly by removing the three outliers in that data set. The data from this algorithm were still widely spread (std. dev. = 0.98). The outliers removed were representative of all images that yielded incorrect results. The first outlier image was presented in Figure 5.3. In the two other outlier

images, some ‘weed patches’ created a bridge between 2 rows, which caused the Similarity Matrix to classify parts of the 2 rows as the same cluster. The angle of the line formed by that cluster wasn’t representative of the angle of the rows. One of the outlier images is presented in Figure 5.4. Clusters #1, 2 and 3 in Figure 5.4 were used to determine the orientation of the inter-row spaces. However, because clusters #1 and 3 included some ‘weedy’ centre points, the resulting stripe angle was in error by approximately  $10^\circ$  (Appendix E.1.1). Cluster #4, which is formed only by ‘weed patch’ centres, is a good example of the robustness of the algorithm against some ‘weed patches’. This cluster wasn’t taken into consideration because it included only 7 centres. The algorithm specified that only clusters of more than 9 centres would be considered, exactly to avoid situations like Figure 5.4. Notice that the smallest ‘weeds’ formed by the sunflower seeds weren’t considered in the linear regression probably because they didn’t meet the minimum number of black pixels (15 pixels) between 2 transitions in the Edge detection. If they did, they probably didn’t have any neighbour, therefore they weren’t considered in the clustering procedure.

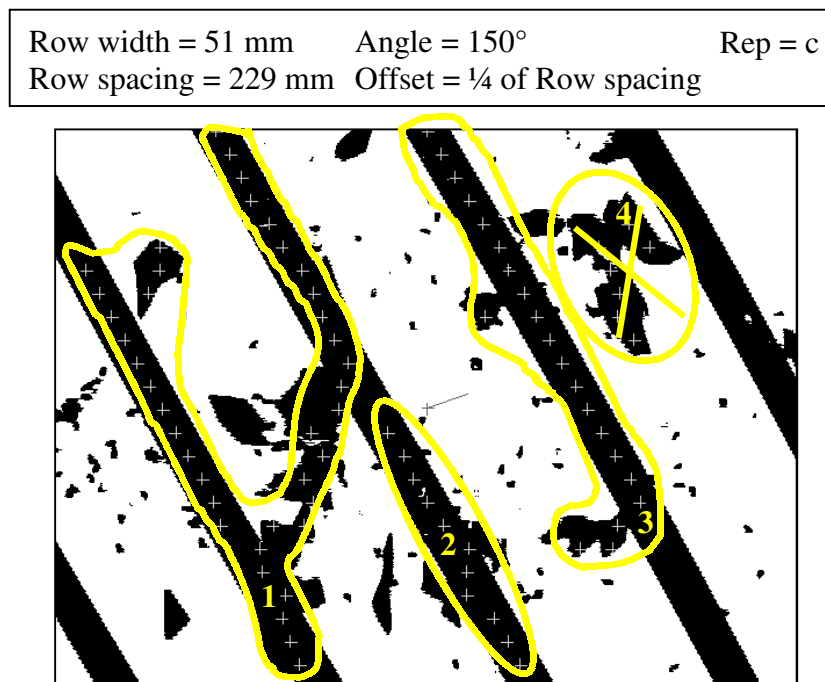


Figure 5.4 Outlier image in the determination of stripe orientation by the Linear Regression algorithm.

Because the Linear Regression and Hough Transform algorithms shared the same steps of edge detection and clustering, the difference in results was likely due to the linear regression itself. The linear regression was performed on every point forming the cluster of a row, regardless of the position of the points. Alternatively, the Hough Transform determined the equation of a line representing the row based on the greatest number of cluster points included on that line. If “weedy points” were clustered by mistake as part of the row, they probably didn’t contribute to the determination of the line equation (Fig. 5.5).

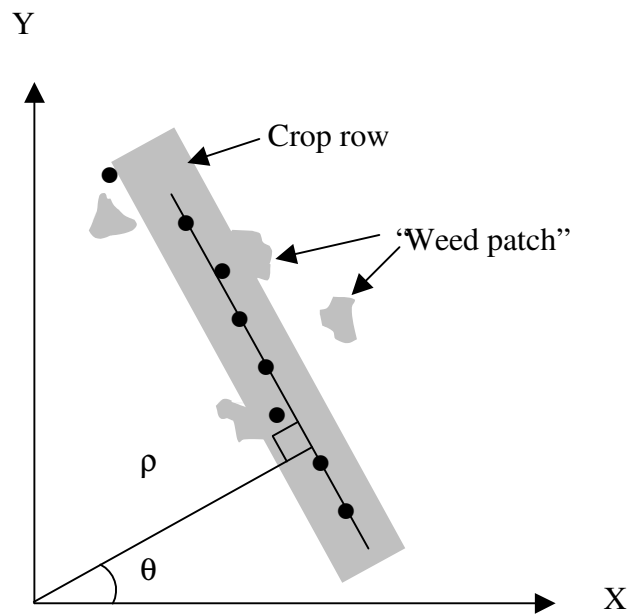


Figure 5.5 Example of cluster where “weedy centre points” didn’t contribute to the determination of the equation of the row with the Hough Transform.

Two outliers were removed from the Stripe Analysis sub dataset, one being the same outlier that was removed from all noise-free data sets (RW = 51 mm, RS = 229 mm, Angle = 150°, Offset = 3/4RS, Rep = c). The second outlier removed is presented in Figure 5.6. “Weed patches” along the edges of the stripe found probably caused the error. Generally, this algorithm performed well in the presence of noise.

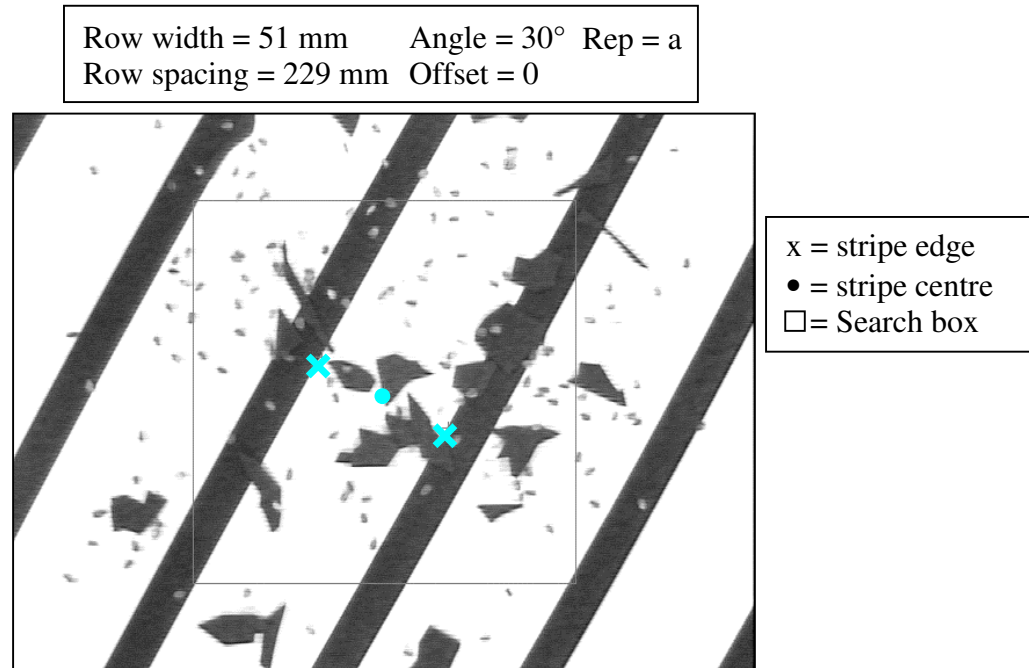


Figure 5.6 Example of noisy image processed with the Stripe Analysis algorithm.

The results of analyses using noise-free versus noisy images were consistent with expectations. The commands from the imaging library used to create the Stripe Analysis algorithm were known from experience to be robust. It was not surprising to see this algorithm performing the same way with or without noise because this noise was randomly scattered in the image and did not nearly resemble the straight edges for which the algorithm was searching. On the other hand, the Blob Analysis algorithm was predictably sensitive to noise. Indeed, the angle of the inter-row spaces, or blobs, was computed from the relative positions of the white pixels forming the blob. If the noise was asymmetrical in the blob, the orientation of the blob's longest axis may have been misleading. This also accounts for the spread of the data with noisy images. The performance of the Blob Analysis algorithm in the field may have depended largely upon the situation, i.e. the location and density of weeds. Published literature (Marchant et al., 1997) suggested that the Hough Transform is tolerant to a small amount of outliers in the linear pattern (noise). With a 73% increase in error mean when noise was added to the images, it could be concluded that the data from this work was not consistent with the literature. However, the second analysis performed without the only outlier in the

data was consistent with the literature because the results were the same regardless of the presence or absence of noise in the images.

ANOVAs were performed on each algorithm's noise-free and noisy data sets, with and without outliers (Appendix E.2.1) to investigate possible interactions among the factors describing the images. Indeed, the ANOVAs suggested the presence of interactions between the factors RW, RS, angle and offset. However, the interactions weren't the same for each algorithm. Further investigation of the data is required to determine the effect of each factor on the results.

### 5.1.2 Determination of stripe location

Means were also calculated for the error in the determination of the location of inter-row spaces for noise-free and noisy images (Table 5.4). Again, an ANOVA was performed on the noise-free and noisy data to investigate differences among the means of each algorithm. The complete ANOVA programs and outputs are presented in Appendix E.2.1.

Table 5.4 Error in determination of the inter-row space location (mm) in original set of data,  $n = 288$ .

Algorithm	Noise-free images		Noisy images		
	Mean	Std dev	Mean	Increase	Std dev
Hough Transform	1.27 <i>a</i>	1.26	6.49 <i>a</i>	411%	17.20
Linear Regression	1.39 <i>a</i>	1.23	8.46 <i>b</i>	509%	16.38
Stripe analysis	2.06 <i>b</i>	1.23	2.11 <i>c</i>	2%	1.28
Blob analysis	2.64 <i>c</i>	1.70	3.28 <i>d</i>	24%	5.30

Note: Means grouped according to the type of image (noise-free or noisy).

Grouping is based on 95% LSD in a non-parametrical analysis.

Increase = ((mean noisy - mean noise-free) / mean noise-free) x 100

Considering the noise-free results presented in Table 5.4, it might be concluded that the best algorithm was the Hough Transform with 1.27 mm of error, on average. Based on a 95% LSD (Appendix E.2.1), the positional error of the Linear Regression algorithm was

not significantly different from the Hough Transform algorithm. The average error for the other algorithms didn't exceed 2.64 mm. Variation in the data was constant in the Hough Transform, Linear Regression and Stripe Analysis algorithms, with an average standard deviation of 1.2 mm. The data of the Blob Analysis were slightly more widely dispersed, with a standard deviation of 1.7 mm.

While the Stripe and Blob Analysis algorithms provided good results with the addition of noise, the Linear Regression and Hough Transform algorithms were dramatically affected. The mean errors of these algorithms increased by 400% to 500%. The Hough Transform algorithm still performed slightly better than the Linear Regression algorithm. Based on the standard deviations, variation in the data was a problem for the Linear Regression and the Hough Transform algorithms. These algorithms presented a variation of 265% and 194% of the mean, respectively. Again, it is observed that the Stripe Analysis algorithm mean error and standard deviation weren't affected by the addition of noise, which made it applicable to use in field conditions.

The poor performances of the Linear Regression and Hough Transform algorithms with noisy images were investigated. The calculation of the RStudent statistic (Appendix E.2.1) revealed the presence of a great number of outliers, both in the noise-free and noisy data sets. Table 5.5 presents the results of the ANOVA without outliers. Tables 5.6 and 5.7 specify which outliers were removed. Again, the multiple comparisons of the lsmeans (95% LSD) were performed in a non-parametrical analysis because the data failed the tests for normality (Appendix E.2.1).

Table 5.5 Error in the determination of the inter-row space location (mm), outliers removed.

Algorithm	Noise-free images			Noisy images			
	<i>n</i>	LSMean	Std dev	<i>n</i>	LSMean	Increase	Std dev
Hough Transform	279	1.14 <i>a</i>	1.02	276	3.51 <i>a</i>	209%	9.35
Linear Regression	280	1.27 <i>a</i>	0.97	278	6.18 <i>b</i>	390%	11.05
Stripe Analysis	280	1.98 <i>b</i>	1.12	284	2.05 <i>a</i>	4%	1.21
Blob Analysis	282	2.56 <i>c</i>	1.61	286	2.89 <i>c</i>	13%	2.49

Note: LSMeans grouped according to the type of image (noise-free or noisy).

Grouping is based on 95% LSD in a non-parametrical analysis.

Increase = ((mean noisy - mean noise-free) / mean noise-free) x 100



Table 5.6 Outliers removed from the Stripe Analysis and Linear Regression algorithms  
data in inter-row space location analysis.

Algorithm	Images	RW mm	RS mm	Angle deg	Offset [0,1,2,3]/4RS	Rep
Stripe Analysis	Noise-free	51	152	150	3	b
		51	229	150	0	b
		51	229	150	0	c
		51	229	150	1	b
		51	229	30	0	a
		76	152	150	2	c
		76	152	150	3	b
		76	229	150	2	b
	Noisy	51	152	150	3	b
		51	229	150	1	b
		76	152	150	2	c
		76	152	150	3	b
Linear Regression	Noise-free	51	152	150	3	b
		51	229	150	0	b
		51	229	150	0	c
		51	229	150	1	b
		76	152	0	0	c
		76	152	150	2	c
		76	152	150	3	b
		76	229	150	2	b
	Noisy	51	152	30	2	a
		51	152	30	3	b
		51	152	60	1	a
		76	152	150	1	a
		76	152	150	3	a
		76	152	30	0	a
		76	152	30	2	a
		76	152	90	2	a
		76	229	60	2	a
		76	229	90	3	a

Note: RW = Row width, RS = Row spacing.

Table 5.7 Outliers removed from the Hough Transform and Blob Analysis algorithms data in inter-row space location analysis.

Algorithm	Images	RW mm	RS mm	Angle deg	Offset [0,1,2,3]/4RS	Rep
Hough Transform	Noise-free	51	152	150	3	a
		51	152	30	2	a
		51	229	150	0	a
		51	229	150	1	a
		76	152	0	0	a
		76	152	150	0	a
		76	152	150	2	a
		76	152	150	3	a
		76	229	150	2	a
	Noisy	51	152	150	0	a
		51	152	150	2	a
		51	152	30	2	a
		51	152	60	1	a
		51	152	60	2	a
		51	229	120	3	a
		76	152	150	1	a
		76	152	30	0	a
		76	152	30	2	a
		76	152	90	2	a
		76	229	60	2	a
		76	229	90	3	a
Blob Analysis	Noise-free	51	152	150	3	a
		51	229	120	2	a
		51	229	150	1	a
		76	152	150	2	a
		76	152	150	3	a
		76	229	120	3	a
	Noisy	76	229	150	1	a
		76	229	90	1	a

Note: RW = Row width, RS = Row spacing.

In noise-free images, the outliers removed were probably caused by a misalignment of the pattern on the rotational platform. This conclusion is partly based on the observation that the same four outliers are present in each algorithm noise-free data set. Incidentally, these four outliers are also present in the noisy images of the Stripe Analysis algorithm

(Table 5.6). Other outliers are repeated among different algorithm's noise-free data sets. Moreover, a visual inspection of the noise-free outlier images didn't allow discerning any specific feature that might have caused the errors. Even though the RStudent statistic affirmed that they were outliers, the errors associated with these outliers were very small. Therefore, the noise-free outlier images are not presented or discussed in the present section.

The mean error and standard deviation of the Blob Analysis algorithm were improved when the 2 major outliers were removed from the noisy data set. One of these outliers (RW = 76 mm, RS = 229 mm, Angle = 90°, Offset = 1/4RS, Rep = c) was also removed in the analysis for the error in row orientation (Fig. 5.2 (b)). The small white blobs were again the reason for the poor positioning result, as this white blob wasn't spread over the whole inter-row space width. The same problem explained the results of the second outlier, presented in Figure 5.7.

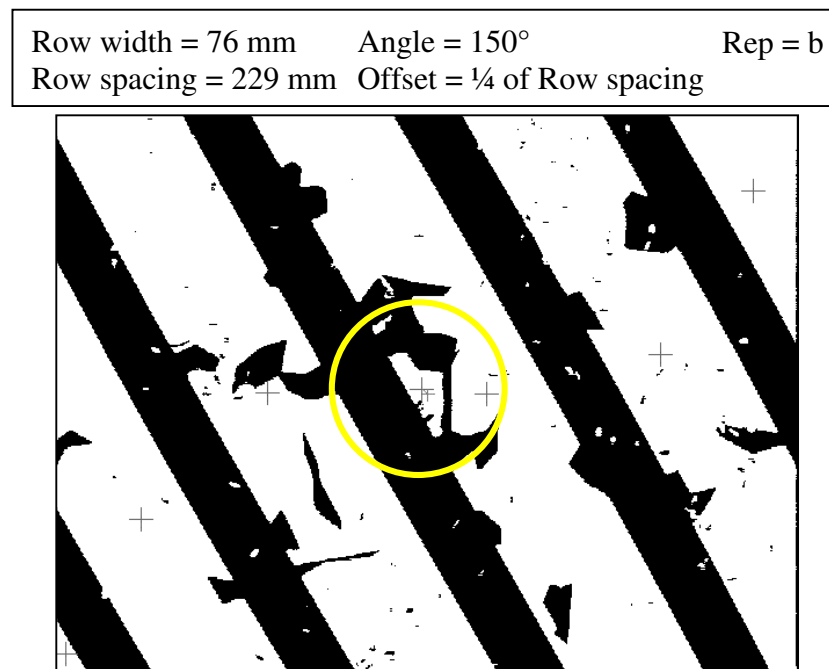


Figure 5.7 Outlier image in the determination of stripe location by the Blob Analysis algorithm.

The reason why the image presented in Figure 5.7 didn't create an erroneous value of row orientation is because the longest axis of the small white blob created by the "weeds" was somewhat parallel to the rows.

The mean error of the Stripe Analysis algorithm didn't significantly decrease with the removal of 4 outliers in the noisy data set. As mentioned before, these outliers were also outliers in the noise-free data sets of the other algorithms. It is then believed that the error in these images wasn't caused by the noise, but by a misalignment of the patterns on the rotational platform.

The mean error of the Hough Transform algorithm using the noisy images decreased by almost 50% with the removal of the 12 outliers, whereas the Linear Regression mean decreased by approximately 25% with the removal of 10 outliers. Table 5.7 indicates that 7 out of 12 Hough Transform algorithm outliers also caused a notable error when processed by the Linear Regression algorithm. In these outlier images, three different problems were defined. In images like Figure 5.8, the problem was similar to Figure 5.3 in Section 5.1.1. The "weed patches" along the row were classified as part of the row and caused the formation of two different clusters for the same row. While this didn't affect the calculation of the angle of the rows, it did affect the determination of the inter-row location because this location was determined from the equations of the 2 clusters closest to the centre of the image.

Row width = 76 mm	Angle = 60°	Rep = a
Row spacing = 229 mm	Offset = ½ of Row spacing	

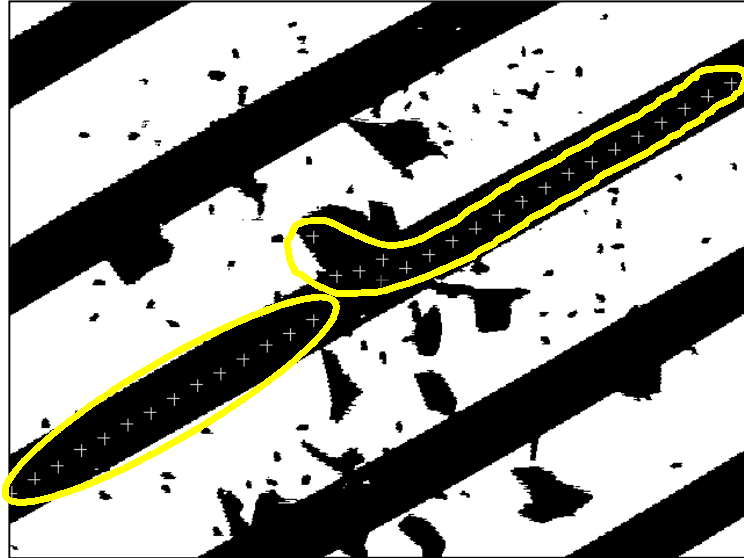


Figure 5.8 Outlier image in the determination of stripe location by the Hough Transform and Linear Regression algorithms.

Further investigation of the data revealed the second challenge faced by the Hough Transform and Linear Regression algorithms in the determination of the inter-row space location. “Weedy” bridges between the rows, causing 2 rows to be classified as one, were the main source of error in the determination of the inter-row space location and orientation. Figure 5.9 presents an example of that situation. Although the image in Figure 5.9 wasn’t classified as an outlier in the Linear Regression algorithm outlier analysis, it did cause an important error in the determination of the inter-row space location.

Row width = 51 mm	Angle = 150°	Rep = b
Row spacing = 152 mm	Offset = ½ of Row spacing	

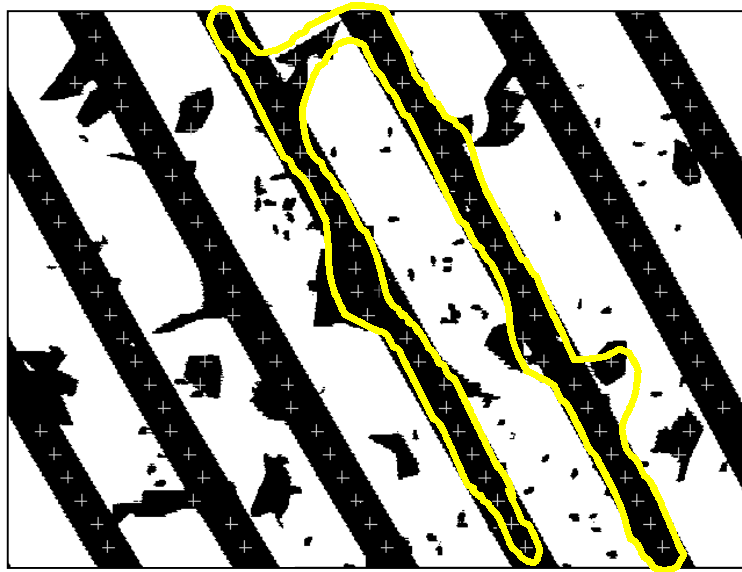


Figure 5.9 Outlier image in the determination of stripe location by the Hough Transform and Linear Regression algorithms.

A third category of images induced positional errors in the Hough Transform and Linear Regression results: images where there were more than 6 clusters found. In images with a RS of 152 mm, up to 7 rows could be distinguished. When designing the algorithms, it was believed that the consideration of 6 rows, or clusters in the image would be sufficient to extract the crop row parameters. That hypothesis was found true in most images. However, in images like Figure 5.10, one of the two most important rows (the ones closest to the image centre) wasn't considered in the calculation because the maximum number of 6 clusters had been reached.

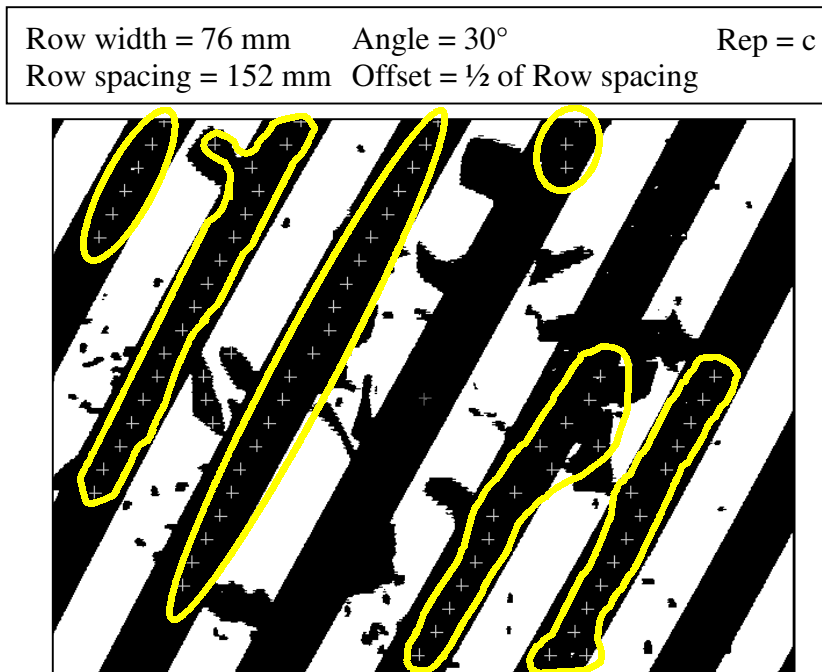


Figure 5.10 Outlier image in the determination of stripe location by the Hough Transform and Linear Regression algorithms where the maximum number of 6 clusters was reached.

Two of the Linear Regression algorithm outliers revealed the weakness of the linear regression compared to the Hough Transform. In these images, some rows were clustered in 2 groups. However, the Hough Transform yielded more accurate results than the linear regression. It is believed that the weedy centre points along the crop rows affected the calculation of the crop row equation with the linear regression, but the Hough Transform process was immune to them. These outlier images are a good example of the robustness of the Hough Transform against “outlier data points” explained in Section 5.1.1. Figure 5.11 presents one of the two outlier images.

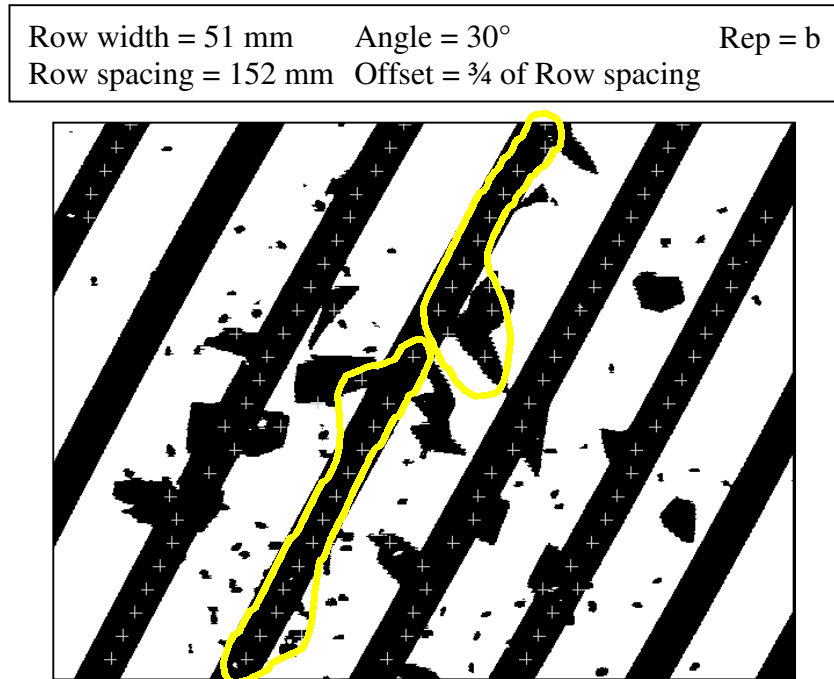


Figure 5.11 Outlier image in the determination of stripe location by the Linear Regression algorithm.

The analysis of the data without outliers revealed that the Stripe algorithm was again the most robust. It provided good results in terms of accuracy and variation in the data, with and without “weed” noise in the images.

In the present project, a positional error of more than 15 mm was not acceptable. The ultimate objective was to be able to identify the centrelines of inter-row spaces and position a sampling tool without harming the plants growing in the rows and to provide good quality data. A positional error of 10% of the row spacing was arbitrarily deemed acceptable. As an example, a camera used to inspect for weeds growing in the inter-row space of crops may be subject to shadows if it is not directly over the centreline of the inter-row space. Excessive positional error could result in images dominated by the shady regions of the inter-row space. All algorithms provided a positional error of less than 15 mm.

ANOVAs were performed on each algorithm’s noise-free and noisy data sets, with and



without outliers (Appendix E.2.1). The ANOVAs suggested the presence of interactions between the factors RW, RS, angle and offset. Further investigation of the data is required to determine the nature of these interactions.

### 5.1.3 Processing time

Table 5.8 presents the average processing time per image for each algorithm. The Stripe Analysis algorithm presented a processing time of approximately 2.3 s per image, with and without noise. In the case of this algorithm, the processing time depended largely on the precision required in angle determination. The algorithm can be faster at the expense of being less accurate (see Section 4.1.2). The Blob Analysis algorithm was fastest, with an average processing time of a third of a second. The algorithm presenting the longest processing time was the Hough Transform algorithm, with an average of 6.34 s per noise-free image and 7.57 per noisy image. The difference between the Linear Regression and Hough Transform algorithms confirmed the fact that the Hough Transform is indeed computationally heavy and slow to process, as mentioned in the literature (Reid and Searcy, 1991). However, processing speed was not considered a critical issue in the present project. Therefore, processing speed should be considered a secondary criterion for choosing the best algorithm.

Table 5.8 Average processing time (s) per image for each algorithm,  $n = 288$ .

<b>Algorithm</b>	<b>Noise-free images</b>	<b>Noisy images</b>
Stripe Analysis	2.31	2.34
Blob Analysis	0.35	0.36
Linear Regression	1.09	1.38
Hough Transform	6.34	7.57

### 5.1.4 Summary of analysis and application of the algorithms

The Stripe Analysis algorithm was considered the best overall. It provided good results in terms of accuracy and variation in the data, with and without “weed” noise in the

images, and its processing time per image was acceptable for the present project. The results of the Blob algorithm were also acceptable, but the algorithm was more sensitive to the presence of weeds. This algorithm was also the fastest. Images where the closest blob of white pixels was a small blob created by the dispersion of the "weeds" were the most problematic. A potential solution to limit the mean positional and orientation errors in such situations would be to increase the minimum number of pixels required to consider a blob in the analysis. This parameter, currently set to 200 pixels, allowed white blobs as small as  $408 \text{ mm}^2$  to be considered as inter-row spaces. The performance of the Linear Regression and Hough Transform algorithms with noisy images was dependent on the specific locations of noise in the image. Positional error could be reduced by allowing more than 6 clusters to be considered in the analysis. This would ensure that all "crop rows" are considered, even in the presence of "weed patches". Because contiguous groups of "weeds" altered the shape of the "crop rows" and caused positioning and orientation errors, it might be useful to apply a filter to the centres found with Edge detection to consider only "true" row centres. A second classification technique could also be used to confirm the Similarity Matrix results. Unfortunately, the processing time per image would increase. All algorithms yielded good results with noise-free images, in terms of determining orientation and location of the inter-row spaces.

The results of the analyses have indicated how these algorithms would perform with images of perfectly straight line patterns and suggest how they may perform under different circumstances. Clearly, images of crop rows won't always be straight, with parallel sharp edges because of the plant leaves, and plants may not easily be segmented from the background, resulting in discontinuous rows in the images. Thus, it's reasonable to expect that additional pre-processing of images, such as dilation and erosion of blobs will be required to successfully implement the Line-detection algorithms described herein, when using field images. The objective of the proposed pre-processing would be to produce images that resemble the noise-free images processed here.

## 5.2 XYZ $\theta$ -table

Figure 5.12 presents an example of the images gathered to evaluate the XYZ $\theta$ -table. The original and processed data related to the testing of the XYZ $\theta$ -table are presented in Appendix E.1.2.

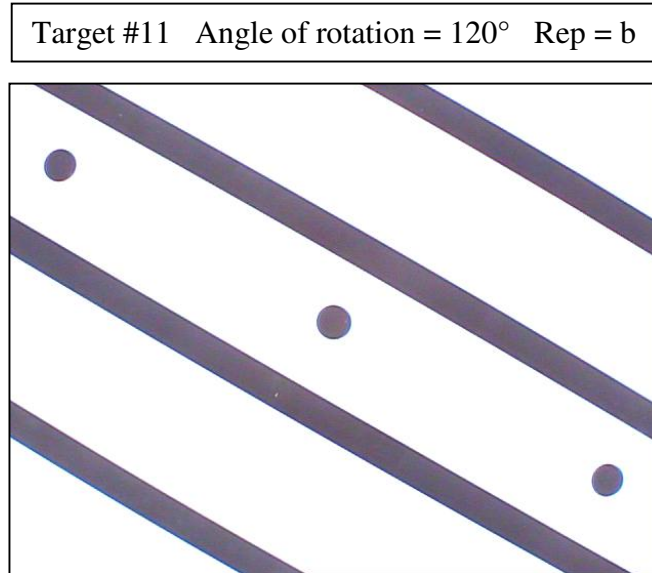


Figure 5.12 Example of image collected to evaluate the XYZ $\theta$ -table.

The ability of the XYZ $\theta$ -table to rotate and position the camera at a specific orientation and location were evaluated.

### 5.2.1 Camera rotation

An ANOVA was performed on the data to determine the mean rotational error and investigate interactions between the two factors in the model: the angle of rotation and the target location. The complete ANOVA programs and outputs are presented in Appendix E.2.2.

The mean error in camera rotation was 0.07°. The ANOVA suggested that there was significant interaction between the angle of rotation and the target location (P-value <

0.001). This could mean that for a particular angle of rotation, the rotational error was different according to the location of the camera on the line pattern, which was surprising. From experience, the  $\theta$ -table supporting the camera was known to be still when it was moved in the XY-space. Montgomery (1991) stated that if an interaction between two factors A and B is significant, the effect of the single factors have little practical meaning, and that in such situation, the effect of factor A must be examined for each level of factor B. Therefore, a histogram quantifying the mean rotational error according to target location and angle of rotation was generated (Fig. 5.13). Each column represents the mean error for 3 replicates defined by a common target location and angle of rotation.

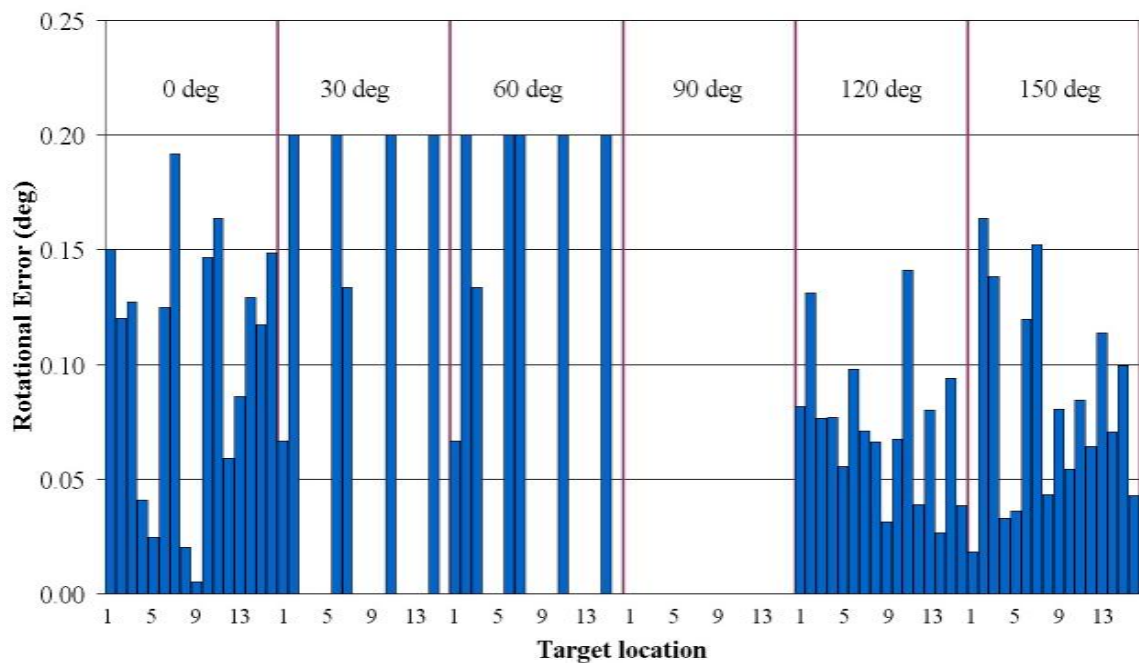


Figure 5.13 Camera rotation error by rotation angle and target location.

In Figure 5.13, it can be observed that the error for all images gathered when the camera was rotated at  $90^\circ$  was 0. All other mean errors ranged from 0 to  $0.2^\circ$ . Because no particular pattern was visible on the graph presenting the alleged interaction, the data are presented again according to the angle of rotation only (Fig. 5.14).

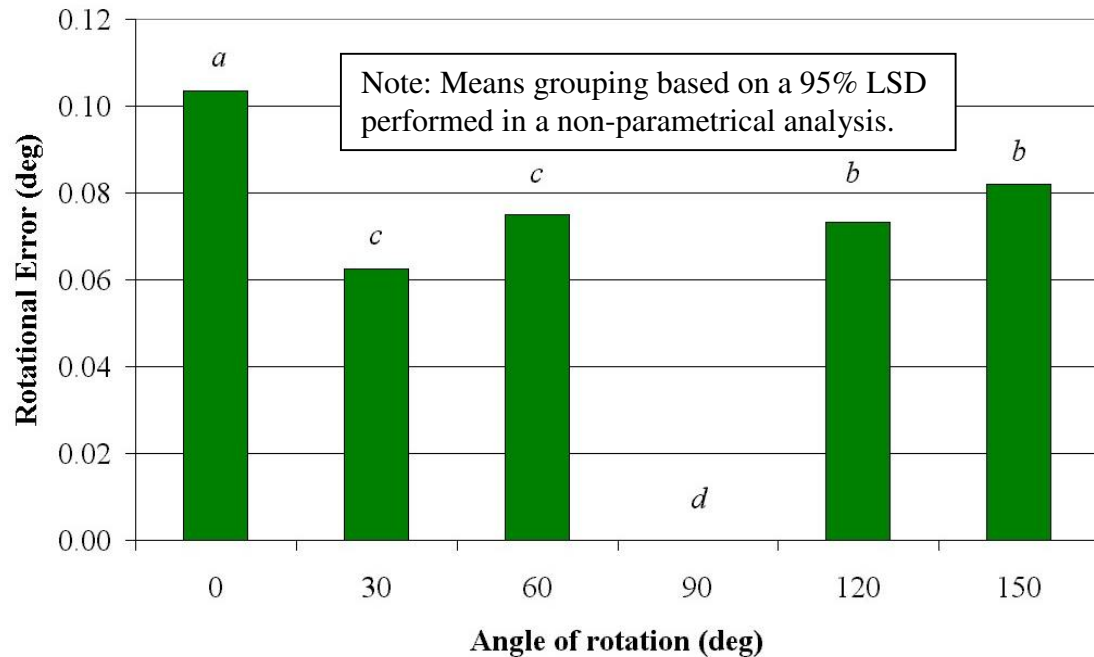


Figure 5.14 Error on camera rotation by rotation angle.

All mean errors were below  $0.12^\circ$ . Tests for normality revealed that the data wasn't normally distributed. Therefore, a non-parametrical analysis of the variance including multiple comparisons on the mean ranks was performed. A 95% LSD (Appendix E.2.2) suggested that the greatest mean error, at the desired orientation of  $0^\circ$ , was significantly different from other means, as was the error mean at  $90^\circ$ . Again, no particular pattern was visible on the graph.

### 5.2.2 Camera positioning

An ANOVA was also performed on the positioning data to determine the mean error and investigate interactions between the angle of rotation and the target location. The complete ANOVA programs and outputs are presented in Appendix E.2.2. The mean positional error was 6.7 mm. The ANOVA suggested that the mean error was influenced by the target location and angle of rotation at the same time, which is realistic. For example, it was suspected that the camera wasn't rotating about the centre of the image. If that were the case, any rotation would induce a positional error. To investigate the

relationship between the angle of rotation and target location, the mean errors of each set of 3 replicates were plotted (Fig. 5.15).

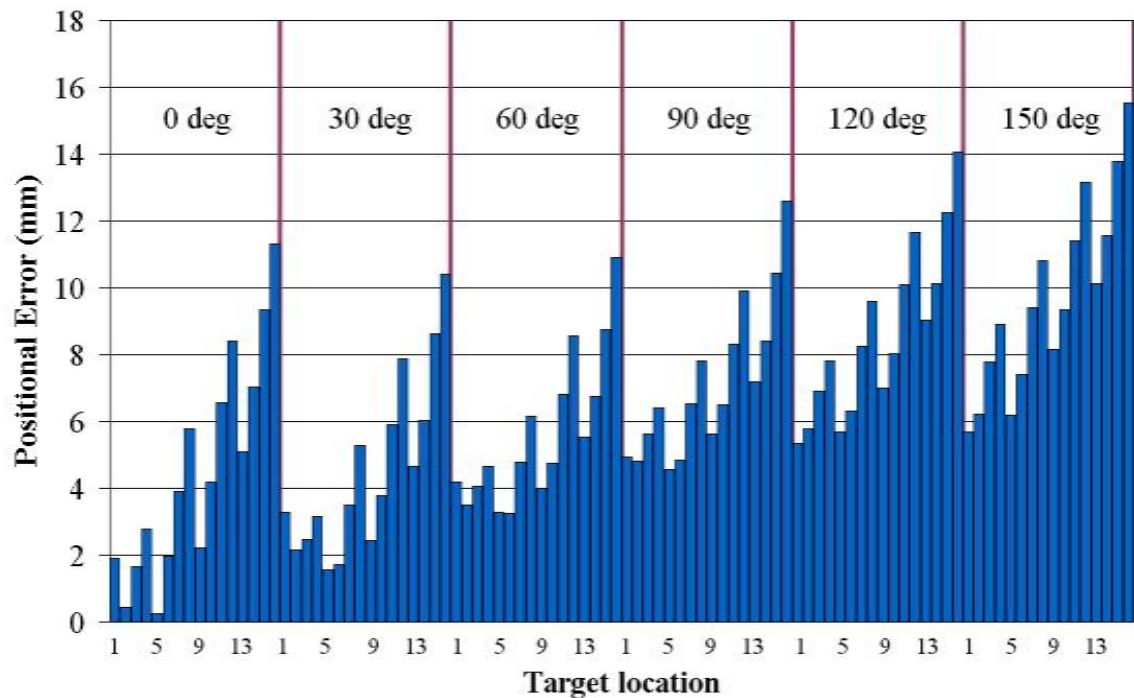


Figure 5.15 Error on camera positioning by angle of rotation and location.

Two conclusions can be made from Figure 5.15. First, the mean positional error increased with the rotation angle. Second, for a particular angle of rotation, the positional error was dependent on the target location. The positional error increased when the camera was moved to location #1 through 4. When the camera was at target #5, the error dropped to a value usually higher than that of target #1, and increased again when the camera was moved to targets #6 through 8, and so on. This particular ‘stairway pattern’ among target locations and the knowledge of the positions of the targets on the line pattern (Fig. 4.14) suggests that the positional error increased as the camera was moved away from the home position along the X-rails and the Y-rail.

To help visualize mean errors, the mean camera location on the line pattern was determined for each set of 3 replicates, by attributing an orientation to the mean errors.

Figure 5.16 summarizes the results. Crosses represent the target locations, and other symbols represent the "mean" location of the centre of the image. The data used to create Figure 5.16 are presented in Appendix E.1.2.

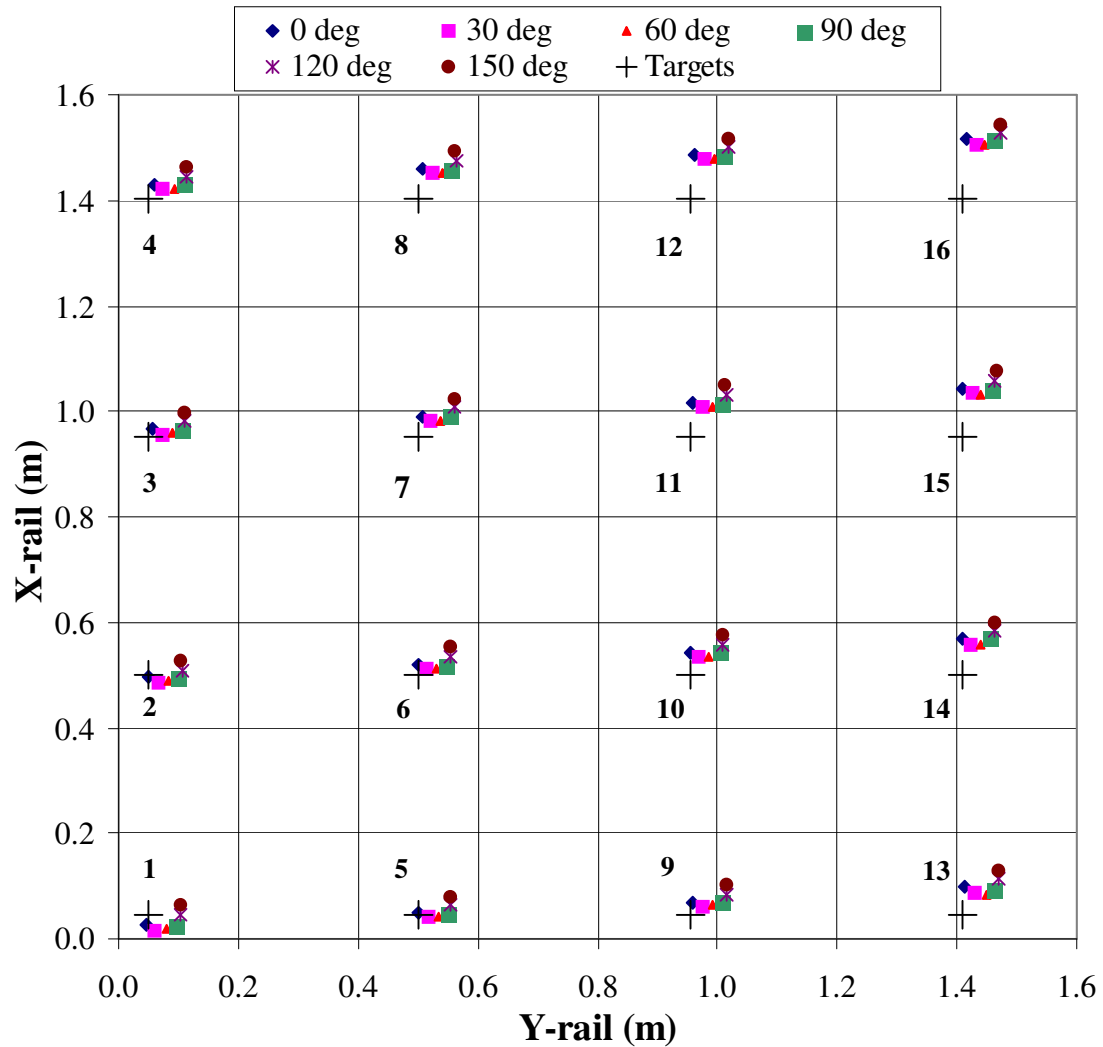


Figure 5.16 Representation of the positional error by target location and rotation angle.

Results presented in Figure 5.16 confirmed some conclusions made from Figure 5.15. One of them was the fact that the positional error increased as the camera was moved away from the home positions (0,0) on the X and Y-rails. As an example, the "cloud" of points at target #16 was far from the location of interest, whereas the centre of the camera was closer to the target when moved to location #1. The error along each rail was quantified. The positional error was expressed as the absolute value of the error

along the X-rails,  $\Delta X$ , and the absolute value of the error along the Y-rail,  $\Delta Y$  (Appendix E.1.2). The mean of  $\Delta X$  was 5 mm, and the mean of  $\Delta Y$  was 3.8 mm, for the entire data set.

More careful investigation of the data was required to find the cause of that increasing error. Figure 5.16 shows that the error along the Y-rail was generally consistent at each rotation angle over all target locations defined by a common location on the X-rails. For example,  $\Delta Y$  at  $0^\circ$  was close to 0 at targets #4, 8, 12 and 16. This means that the displacement along the Y-rail was accurate, but that the calibration of the target locations relative to the home position included an error that reflected in the results at each target. However, a different situation occurred for the X-rails. For a specific location on the Y-axis, represented by a column of 4 targets in Figure 5.16,  $\Delta X$  increased with the displacement of the camera along the X-rails. This was probably due to an error when calibrating the distance travelled along the rail for a certain number of steps of the motor.

For a specific location on the X-axis, represented by a row of 4 targets in Figure 5.16,  $\Delta X$  also increased as the camera was moved along the Y-rail. This was probably due to a mechanical error in the construction of the XYZ $\theta$ -table. Specifically, the Y-rail orientation wasn't perfectly perpendicular to the X-rails. Another possible cause would be that one end of the Y-rail was moving faster than the other. To verify that hypothesis, the average location of the camera was calculated at all targets. Then, a linear regression was performed on the mean camera location for all four groups of targets defined by a common location on the X-rails. The resulting lines were representative of the orientation of the Y-rail at different locations along the X-rails. The slopes of the lines were used to determine the angle between the Y-rail and the theoretical Y-axis. Figure 5.17 presents the results.



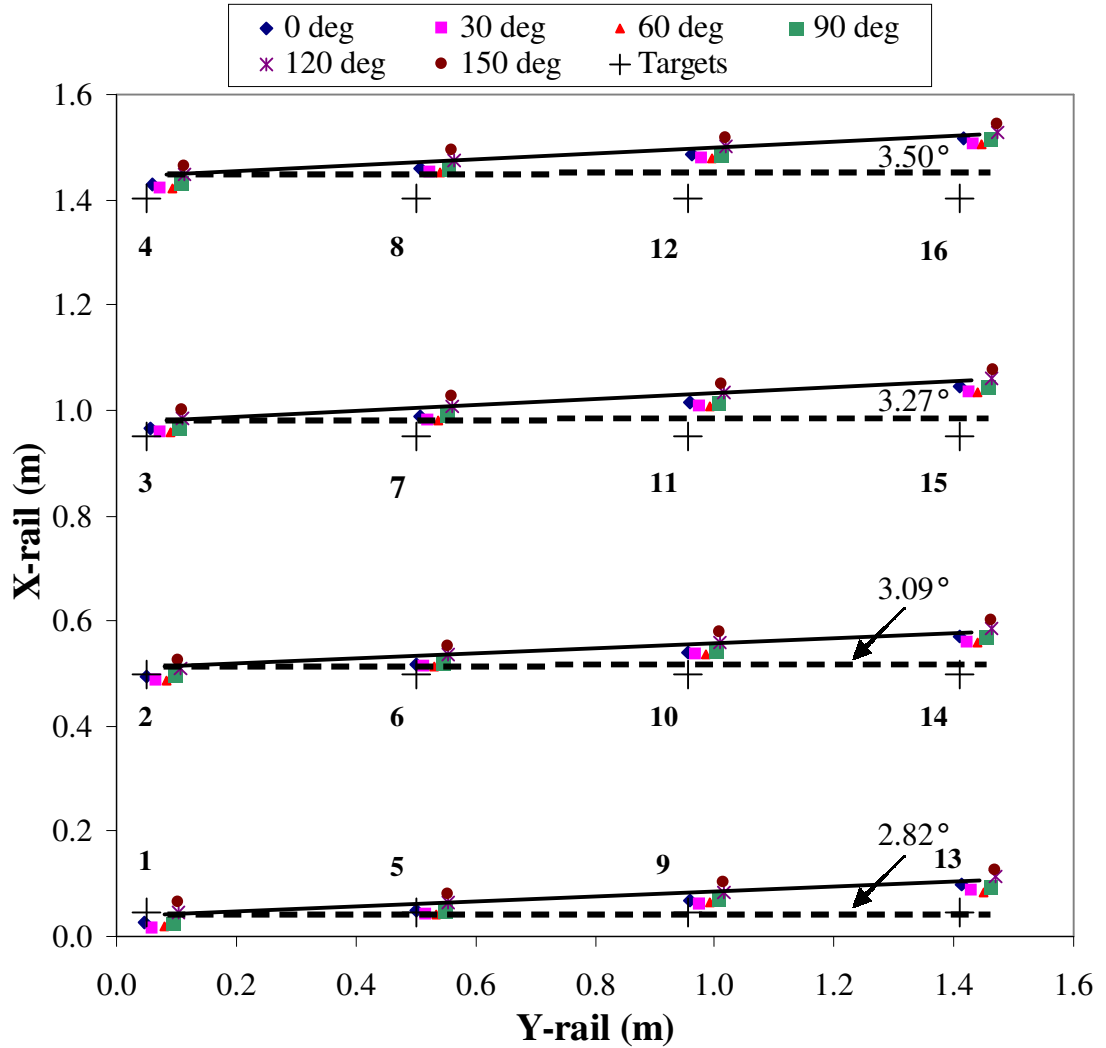


Figure 5.17 Orientation of the Y-rail at different locations on the X-rails.

From Figure 5.17, it is observed that the angle of the Y-rail relative to the theoretical Y-axis was increasing as the Y-rail was moved farther away from the home position on the X-rails. This suggests that the end of the Y-rail farthest from the home position was moving faster than the other end. Slight differences in the pitch of the ball screws might have caused such error. Another cause might be the total tolerance in the displacement of both ends of the Y-rail. Appendix F presents the equation that could be used to calculate that tolerance. Finally, some adjustment in the timing belt might also be required to obtain a perfectly synchronized displacement on both X-rails.

At each location, the mean errors as a function of increasing angle presented the same curvilinear pattern. The centre of the arc formed by the camera centre points wasn't located on the target location. Clearly, the centre of rotation of the camera didn't correspond with the centre of the image, which probably means that the camera was tilted, off centre relative to the axis of the  $\theta$ -table, or both at the same time. Figure 5.18 explains the first two situations.

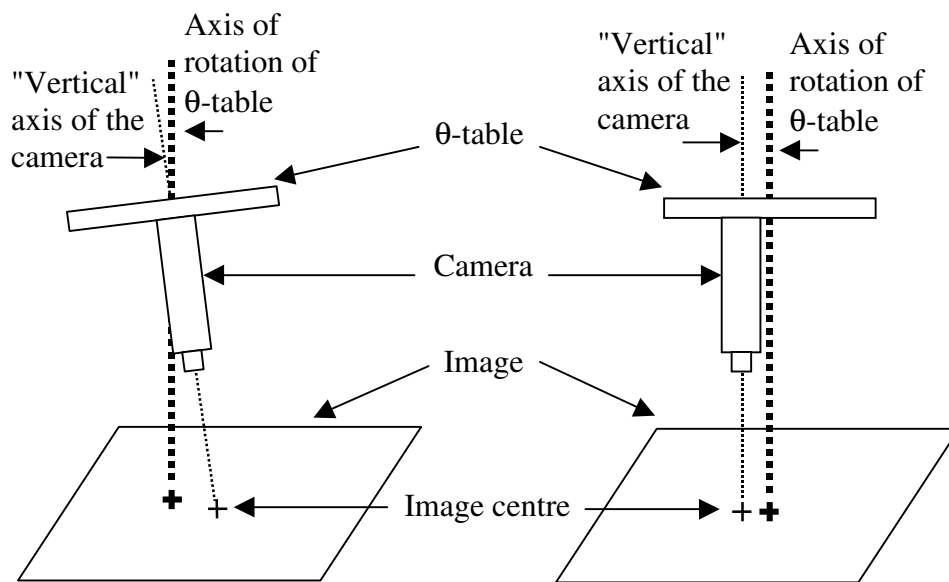


Figure 5.18 Possible causes of positional error induced by the rotation of the camera. (a) camera tilted, (b) camera off centre.

## **6. CONCLUSION**

The purpose of this project was to develop a camera-positioning system for an autonomous weed scout robot. Specifically, the objectives were to develop a vision system that could determine the location and orientation of artificial crop rows in an image, and to build a robotic device that could position a camera and rotate it at a desired location and angular orientation. The first objective was fulfilled by the creation of four Line-detection programs based on different image processing or pattern recognition methods. The second objective was achieved by the construction of an XYZ $\theta$ -table. The performance of the Line-detection programs and the XYZ $\theta$ -table were evaluated.

### **6.1 Line-detection algorithms**

The four Line-detection algorithms were based respectively on a stripe analysis, a blob analysis, a linear regression and the Hough Transform. The first two used functions of a commercial imaging library, whereas the last two determined the equation of the lines representing the rows after detecting the rows from their edges.

The ability of the programs to determine the orientation of a set of simulated crop rows in an image was evaluated. All algorithms behaved approximately the same when determining the rows' angle in the noise-free images, with an average error of 0.5°. Weed-simulating noise was added to the images, which were then reprocessed by the Line-detection programs. The analysis showed that the mean error in orientation and standard deviation increased when the noise was added to the images, up to 1.1° for the Linear Regression algorithm. An important increase was also observed in standard deviation. However, the Stripe Analysis algorithm results weren't affected by the

presence of noise.

The programs were also evaluated based on their ability to determine the location of a point on the centreline of an inter-row space. In noise-free conditions, all algorithms could find the centreline of an inter-row space within 2.7 mm. Again, when the noise was added to the images, the mean error and standard deviation increased for all programs except the Stripe Analysis algorithm. In the case of the Hough Transform and Linear Regression algorithm, the mean error increased to 6.5 and 8.5 mm, respectively.

Further analysis of the data, including outlier analyses, allowed determining several causes of the increase in mean error and spread of the data when the algorithms attempted to determine the orientation of the rows and location of an inter-row space. In the case of the Blob Analysis algorithm, images where the disposition of the weeds created a sub-region within an inter-row space were often the cause of great errors. In the case of the Linear Regression and the Hough Transform algorithms, bridges of weeds between two adjacent crop rows, or weeds in the inter-row space giving the illusion of a crop row were the most misleading. The results improved significantly when the analysis was performed on the data without the outliers.

The mean processing time per image was calculated for each program. The Blob Analysis algorithm was the fastest with a mean processing time of 0.3 s, and the Hough Transform algorithm was the slowest, with a mean processing time of 6.3 s.

Overall, the Stripe Analysis algorithm was considered the best because it was the least affected by the presence of noise in the images, followed by the Blob Analysis algorithm. The performance of the Blob Analysis, the Linear Regression and the Hough Transform algorithms could be improved by some modifications in the programs.

## **6.2 XYZ $\theta$ -table**

The ability of the XYZ $\theta$ -table to accurately move a camera within the XY-space and

rotate it at a desired angle was evaluated in a laboratory setting. It was determined that XYZ $\theta$ -table was able to rotate the camera with a mean error of 0.07°. However, the results suggested that the rotation wasn't centered. It was suspected that the camera wasn't vertical, or that its centre of rotation didn't correspond to the centre of the image.

The mean error in positioning was 7 mm. However, the results suggested that the mechanical construction wasn't perfect. It seems that the rails in the X and Y directions weren't perpendicular. There was an increasing positioning error along the X-rails. Finally, the results suggested that the movement of the two ends of the Y-rail wasn't synchronized. Some simple mechanical adjustments were proposed to improve the results in positioning. Overall, the results in rotation and in positioning were acceptable for the desired applications.

## **7. RECOMMENDATIONS AND FUTURE WORK**

The following recommendations can be made to improve the performance of the Line-detection algorithms.

- (1) For applications requiring a high accuracy and small variance over a broad range of field images (noise-free and noisy images), the Stripe Analysis algorithm should be selected to process the images.
- (2) For applications requiring fast processing, the Blob Analysis algorithm should be used.
- (3) The addition of a pre-processing step aiming at removing most of the noise in the image, and the inclusion of up to 9 clusters in the analysis would improve the performance of the Linear Regression and Hough Transform algorithms with noisy images. Stronger clustering methods or the addition of a second clustering step to confirm the results of the Similarity Matrix would also improve the algorithms performance.
- (4) Future work involves evaluating the Line-detection algorithms with images of living plants, in different lighting conditions.

The following recommendations can be made for future work involving the XYZ $\theta$ -table.

- (1) Reinstallation of the Y-rail perpendicularly to the X-rails may be necessary to achieve positional accuracy.
- (2) The calibration of the X-rails travelled distance per motor step needs to be confirmed.
- (3) Proper care should be given to timing both X-rails together.
- (4) To achieve positional accuracy in rotation, it should be ensured that the central

axis of the camera is vertical.

## REFERENCES

- Allegro Microsystems Inc. 2004. <http://www.allegromicro.com/> Accessed February 2, 2004. Worcester, MA
- Åstrand, B. and A. J. Baerveldt. 1999a. Mechatronics in agriculture – robust recognition of plant rows. In *The 2nd International Conference on Recent Advances in Mechatronics ICRAM'99*, 135-141. Istanbul, Turkey. May 24-26.
- Åstrand, B. and A. J. Baerveldt. 1999b. Robust tracking of plant rows using machine vision. In *The Sixth International Conference on Mechatronics and Machine Vision in Practice*, 95-101. Middle East Technical University, Ankara, Turkey. September 1-3.
- Åstrand, B. and A. J. Baerveldt. 2001. Design of an agricultural mobile robot for mechanical weed control. In *Proceedings of the Fourth European Workshop on Advanced Mobile Robots Eurobot'01*, 139-146. Lund, Sweden. September 19-21.
- Åstrand, B. and A. J. Baerveldt. 2002. An agricultural mobile robot with vision-based perception for mechanical weed control. *Autonomous Robots* 13(1):21-35.
- Åstrand, B. and A. J. Baerveldt. 2003. A mobile robot for mechanical weed control. *International Sugar Journal* 105(1250):89-95.
- Bak, T. 2001. Vision-GPS fusion for guidance of an autonomous vehicle in row crops. In *ION GPS 2001*, 423-451. Salt Lake City, UT. September 11-14.
- Billingsley, J. and M. Schoenfisch. 1995. Vision-guidance of agricultural vehicles.



*Autonomous Robots* 2:65-76.

Billingsley, J. and M. Schoenfisch. 1997. The successful development of a vision guidance system for agriculture. *Computers and Electronics in Agriculture* 16:147-163.

Davies, E.R. 1997. *Machine Vision: Theory, Algorithms, Practicalities*. New York, NY: Academic Press Inc.

Duda, R. O. and P.E. Hart. 1972. A generalized Hough Transformation for detecting lines in pictures. *Communications of the ACM* 15:1-15.

Lintech Company. 2004. Catalogue and <http://www.lintechmotion.com/> Accessed February 9, 2004. Monrovia, CA.

Marchant, J. A., T. Hague and N. D. Tillett. 1997. Row-following accuracy of an autonomous vision-guided agricultural vehicle. *Computers and Electronics in Agriculture* 16:165-175.

Measurement Computing. 2004. <http://www.mccdag.com/> Accessed February 2, 2004. Middleboro, MA

Meisel, W. S. 1972. *Computer-Oriented Approaches to Pattern Recognition*. New York, NY: Academic Press Inc.

Montgomery, D. C. 1991. *Design and Analysis of Experiments*. New York, NY: John Wiley and Sons Inc.

Nielsen, K. M., P. Andersen, T.S. Pedersen, T. Bak and J. D. Nielsen. 2002. Control of an autonomous vehicle for registration of weed and crop in precision agriculture. In *IEEE Conference on Control Applications CCA/CACSD*, 909-914. Glasgow, Scotland. September 18-20.

Palmer, R. and D. Wild. 2000. Motorized weed scouting robot: feasibility and design. Indian Head, SK: Indian Head Agricultural Research Foundation.

Pedersen, T. S., K. M. Nielsen, P. Andersen, T. Bak and J. D. Nielsen. 2002. Development of an autonomous vehicle for weed and crop registration. In *AgEng-2002*, 02-AE-005. Budapest, Hungary. June 30 – July 4.

Reid, J. F. and S. W. Searcy. 1991. An algorithm for computer vision sensing of a row crop guidance directrix. *SAE 1991 Transactions* 100(2):93-105.

Rosenfeld, A. and A. C. Kak. 1976. *Digital Picture Processing*. New York, NY: Academic Press Inc.

SAS Institute Inc. 1999. *SAS/STAT User's Guide, version 8*. SAS Institute Inc., Cary, NC.

Slaughter, D. C., P. Chen and R. G. Curley. 1997. Computer vision guidance system for precision cultivation. ASAE Paper No. 97-1079. St. Joseph, MI: ASAE. 18 pp.

Sørensen, C. G., H. J. Olsen, A. P. Ravn and P. Makowski. 2002. Planning and operation of an autonomous vehicle for weed inspection. ASAE Paper No. 02-1177. St. Joseph, MI: ASAE. 10 pp.

Southall, B., T. Hague, J. A. Marchant and B. F. Buxton. 2002. An autonomous crop treatment robot: part I. A Kalman Filter model for localization and crop/weed classification. *The International Journal of Robotics Research* 21(1):3-16.

Weeder Technologies. 2004. <http://www.weedtech.com/> Accessed February 2, 2004. Ft Walton Beach, FL.

## APPENDIX A LINE PATTERNS USED IN ALGORITHMS TESTING

Figures A.1 to A.4 present the line patterns used to gather the images for the testing of the Line-detection program.

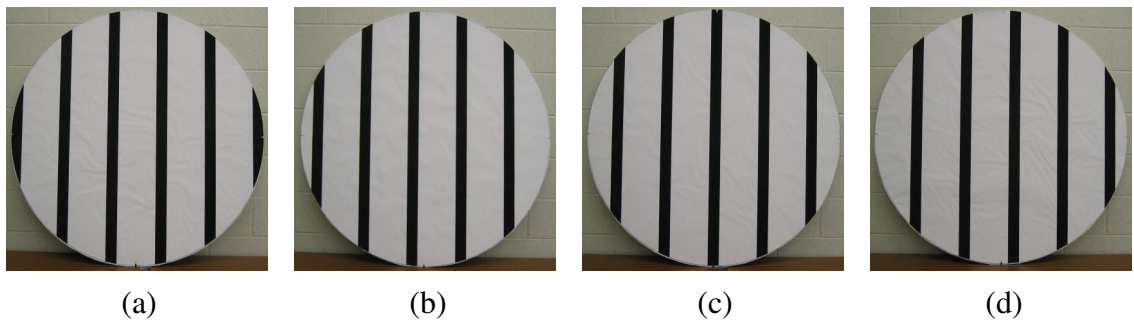
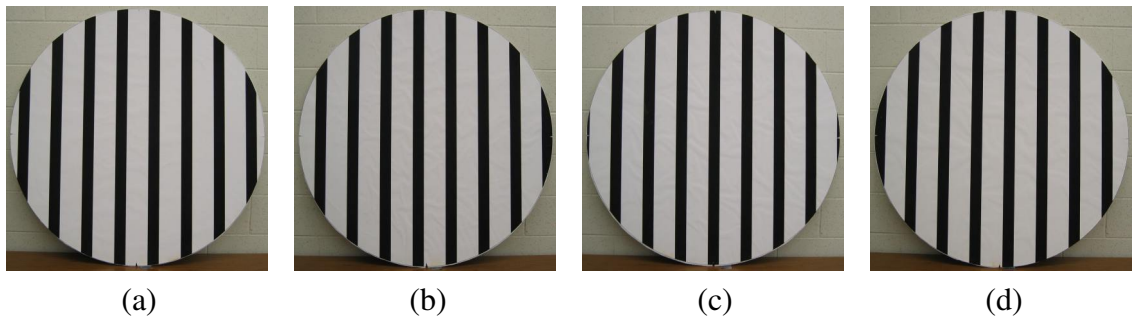


Figure A.2 Line patterns used in Line-detection algorithms testing. Row width = 51 mm, Row spacing (RS) = 229 mm. (a) Offset = 0, (b) Offset =  $\frac{1}{4}$  of RS, (c) Offset =  $\frac{1}{2}$  of RS, (d) Offset =  $\frac{3}{4}$  of RS.

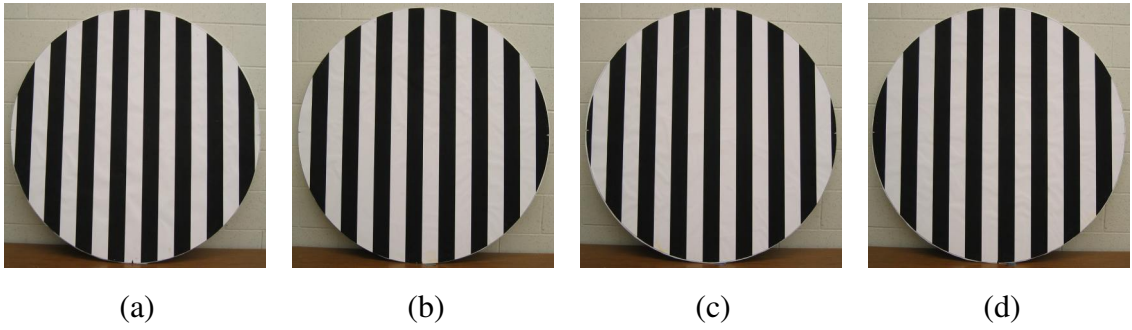


Figure A.3 Line patterns used in Line-detection algorithms testing. Row width = 76 mm, Row spacing (RS) = 152 mm. (a) Offset = 0, (b) Offset =  $\frac{1}{4}$  of RS, (c) Offset =  $\frac{1}{2}$  of RS, (d) Offset =  $\frac{3}{4}$  of RS.

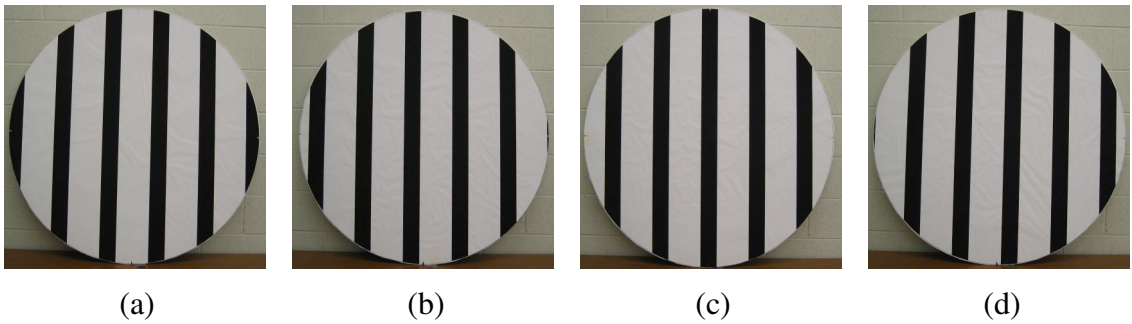


Figure A.4 Line patterns used in Line-detection algorithms testing. Row width = 76 mm, Row spacing (RS) = 229 mm. (a) Offset = 0, (b) Offset =  $\frac{1}{4}$  of RS, (c) Offset =  $\frac{1}{2}$  of RS, (d) Offset =  $\frac{3}{4}$  of RS.

## **APPENDIX B      LINE-DETECTION PROGRAM**

This section presents the Line-detection program. The complete interface of the program was presented in Figure 4.3. To see the code and detailed interface of the Line-detection program, refer to the file LineDetection.vbp in the folder \Appendix B\VB Line-Detection Project\ included on the accompanying CD-ROM. This file was created using Microsoft Visual Basic 6.0 (Microsoft Corporation, Redmond, WA). To facilitate comprehension, comments were inserted in the code. It must be reminded that the program LineDetection.vbp can't be ran without the appropriate framegrabber (Meteor RGB, Matrox Electronics Ltd, Dorval, QC) being installed on the computer.

## APPENDIX C EQUATIONS OF INTER-ROW SPACE CENTRELINES

In order to determine the error in location of the inter-row space centreline, the equations of the inter-row spaces were defined based on each image row spacing, angle, and offset. Two points were defined on the theoretical equation of the inter-row space centreline passing through the centre of the images characterized by an angle of  $90^\circ$  and an offset of 0 (Fig. C.1). Then, from the knowledge of the row spacing, these points were translated along the Y-axis of the image. Each inter-row space in the image was then characterized by 2 points (Fig. C.1).

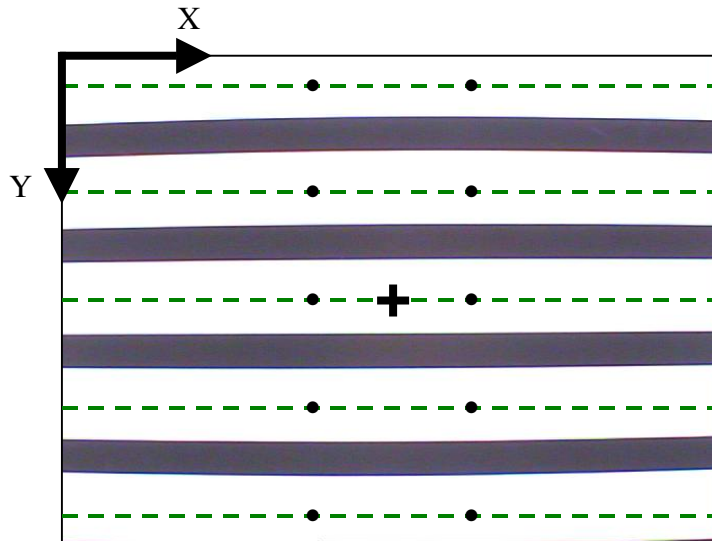


Figure C.1 Points forming the equations of the inter-row spaces in images characterized by a row spacing of 152 mm, an angle of  $90^\circ$  and an offset of 0.

To determine the equation of the line going through each set of two points, vector algebra was used. The line between 2 points is defined as:

$$L = V \times P = \begin{vmatrix} I & J & K \\ v_x & v_y & 1 \\ \rho_x & \rho_y & 1 \end{vmatrix} = (v_y - \rho_y)I + (v_x - \rho_x)J + (v_x \rho_y - v_y \rho_x)K \quad (C.1)$$

where:

$L$  = vector defining the line

$V$  = vector defining the first point

$P$  = vector defining the second point

The resulting equation of the line is then given by:

$$ax + by + c = 0 \quad (C.2)$$

where:

$$a = (v_y - \rho_y)$$

$$b = (v_x - \rho_x)$$

$$c = (v_x \rho_y - v_y \rho_x)$$

The last step was to calculate the perpendicular distance  $d$  from the point  $(x_0, y_0)$  determined by the algorithm as being on the centreline of an inter-row space to each actual inter-row space centreline. This was accomplished using equation (C.3):

$$d = \left| \frac{ax_0 + by_0 + c}{\sqrt{a^2 + b^2}} \right| \quad (C.3)$$

The smallest distance was considered as the error in determination of the inter-row space location. This error, expressed as pixels, was converted to millimetres.

This procedure was also applied for images characterized by an angle of  $90^\circ$ , an offset of 0, and a row spacing of 229 mm. For images presenting a different offset, the original

points used to determine the equation of the inter-row spaces in images presenting an offset of 0 were translated along the Y-axis of the number of pixels corresponding to the offset value of the image. Again, the same procedure was applied to determine the equation of the inter-row spaces.

For images described by common values of offset and row spacing, but presenting an angle different than  $90^\circ$ , the original points used to determine the equation of the inter-row spaces in images at  $90^\circ$  were rotated about the centre of the image using a rotation matrix. The first step was to determine the coordinates of the original points in a  $X'Y'$  coordinate system centered in the image and having the same orientation as the  $XY$  coordinate system (Fig. C.2). This was accomplished by subtracting 320 pixels to the  $x$  coordinate of the points, and subtracting 240 pixels to the  $y$  coordinate.

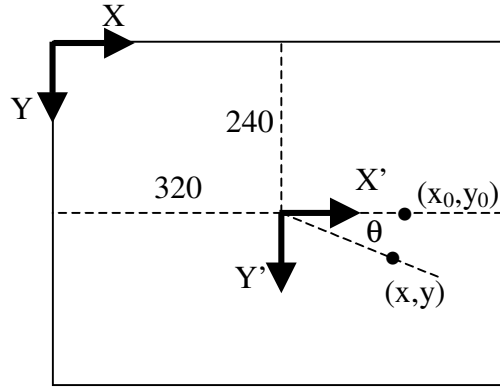


Figure C.2 Rotation of a point about the centre of the image.

The angle of the stripes in the image was then used to perform the rotation of the points about the origin of  $X'Y'$ . The coordinates of the rotated point are given by:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \quad (\text{C.4})$$



where:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \text{vector of the point resulting from the rotation}$$

$\theta$  = angle of rotation from axis X' (increasing clockwise), or angle of stripes -  $90^\circ$

$$\begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \text{vector of a point in the X'Y' coordinate system}$$

The new points resulting from the rotation were used to calculate the equations of the lines representing the inter-row spaces.

## APPENDIX D DESCRIPTION OF THE XYZ $\theta$ -TABLE

This section presents the XYZ $\theta$ -table in details. Pictures of the main parts are presented, as well as specifications of the material used.

### D.1 Specifications of XYZ $\theta$ -table

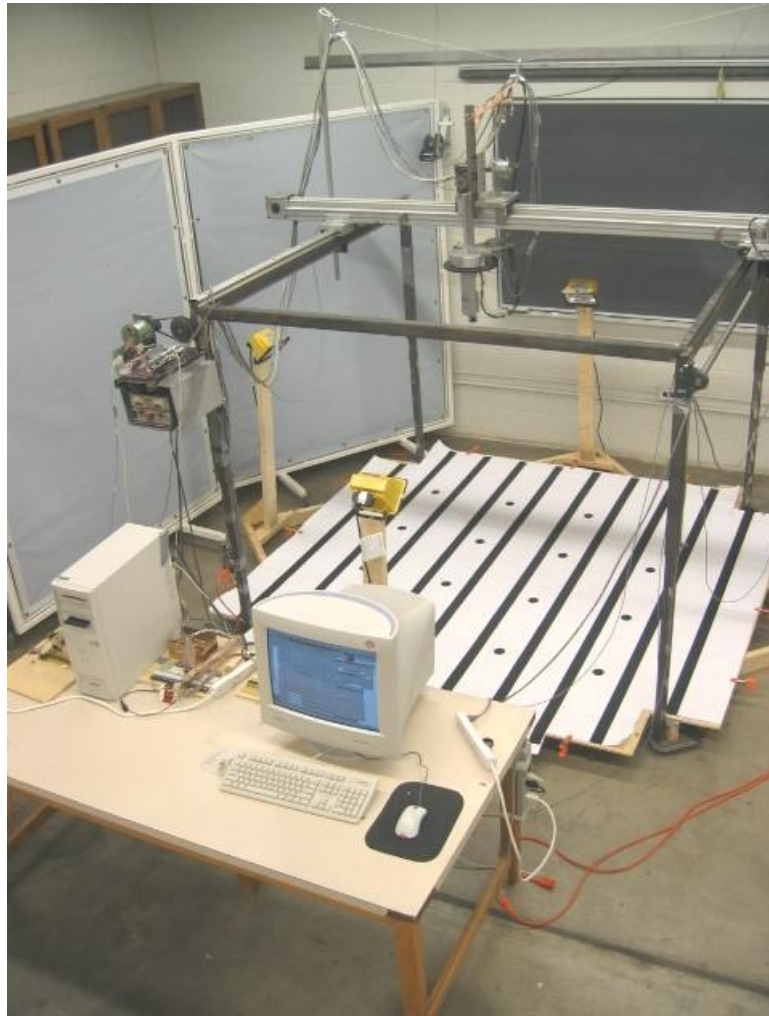


Figure D.1 XYZ $\theta$ -table and control system.

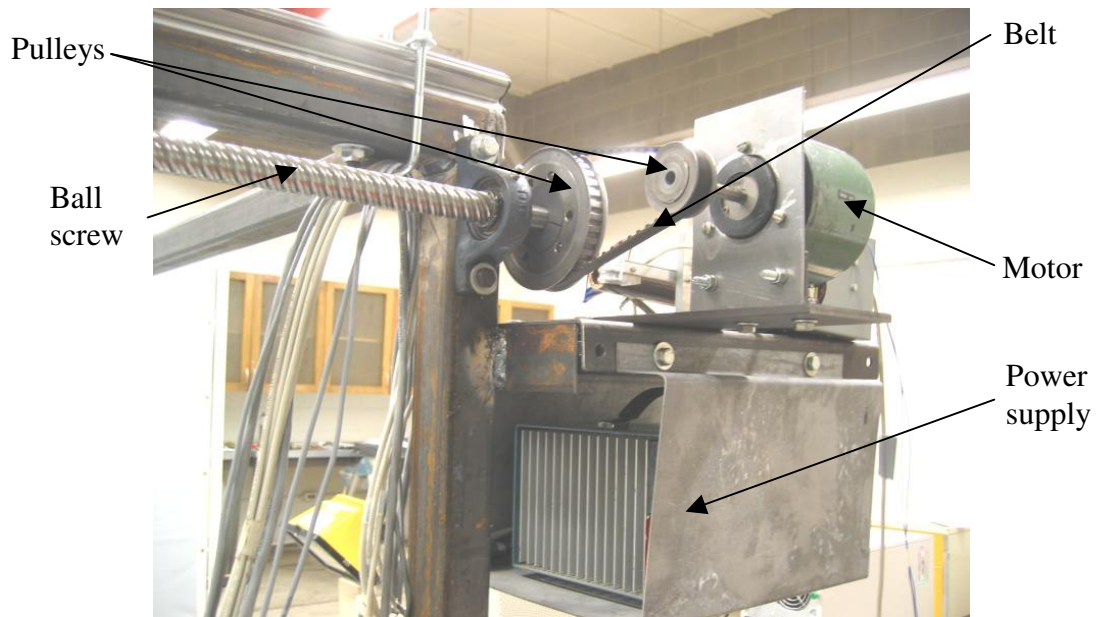


Figure D.2 Motor, power supply and ball screw driving the Y-rail on the X-rails.

Table D.1 Specifications of the motor on the X-rails.

Part name	Stepper Motor
Manufacturer	Sigma Instruments, Braintree, MA
Model	21-4233D-23992
Temperature Rise	50° C @ 25 W
Voltage	23 VDC
Current Draw	0.66 A/phase
Step Angle	1.8°

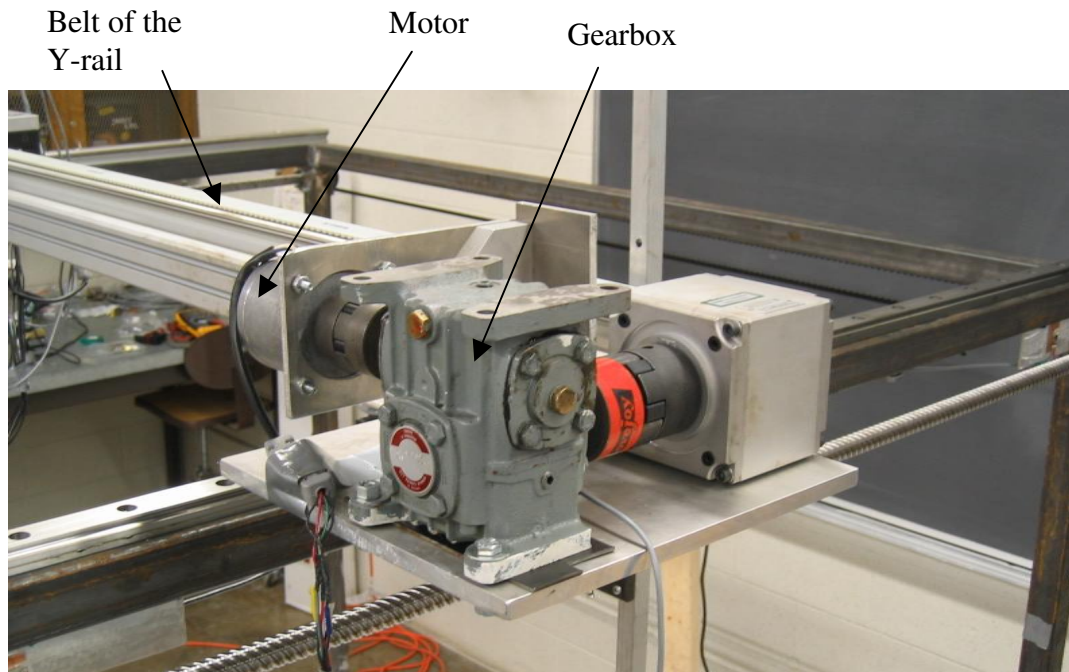


Figure D.3 Motor actuating the belt of the Y-rail and gearbox.

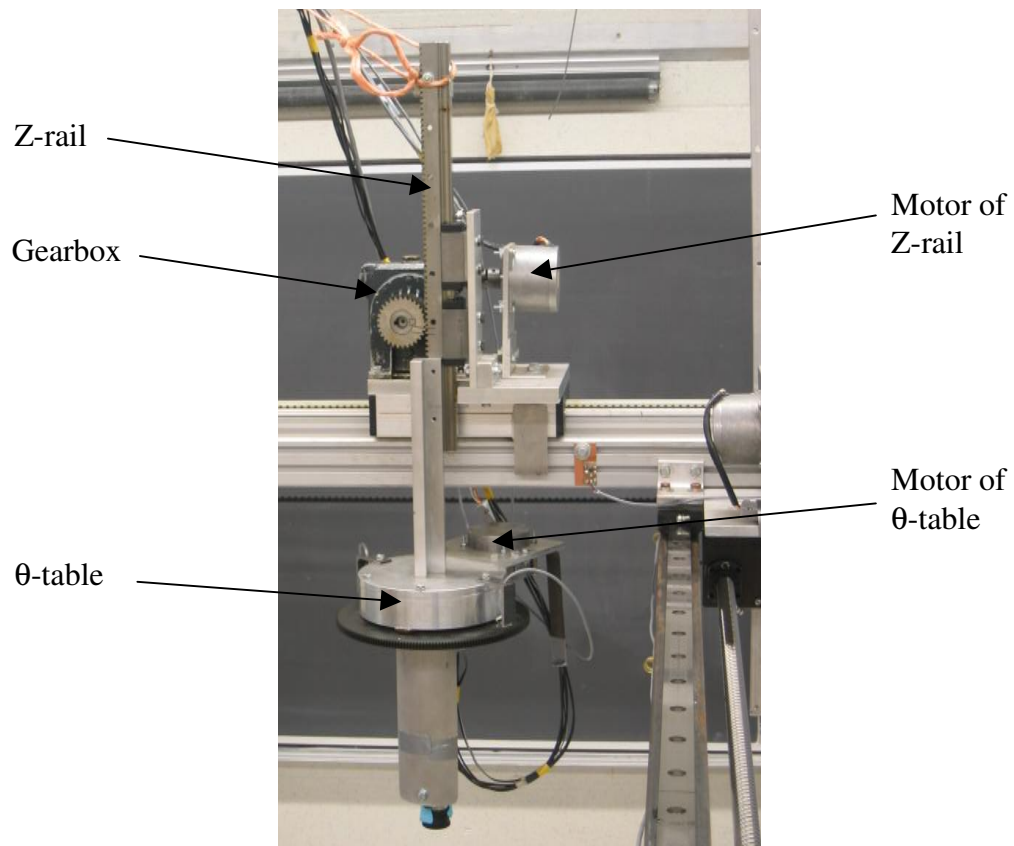


Figure D.4 Z-rail and  $\theta$ -table.

Table D.2 Specifications of the motor on the rails Y and Z, and on the  $\theta$ -table.

Part name	Fuji Step Motor
Part Number	2526734
Manufacturer	Fuji Electric Co., Tokyo, Japan
Model	GPF 3945-2A
Voltage	4.5 VDC
Current Draw	1.4 A/phase
Step Angle	2°
Class	B

## D.2 Controls hardware

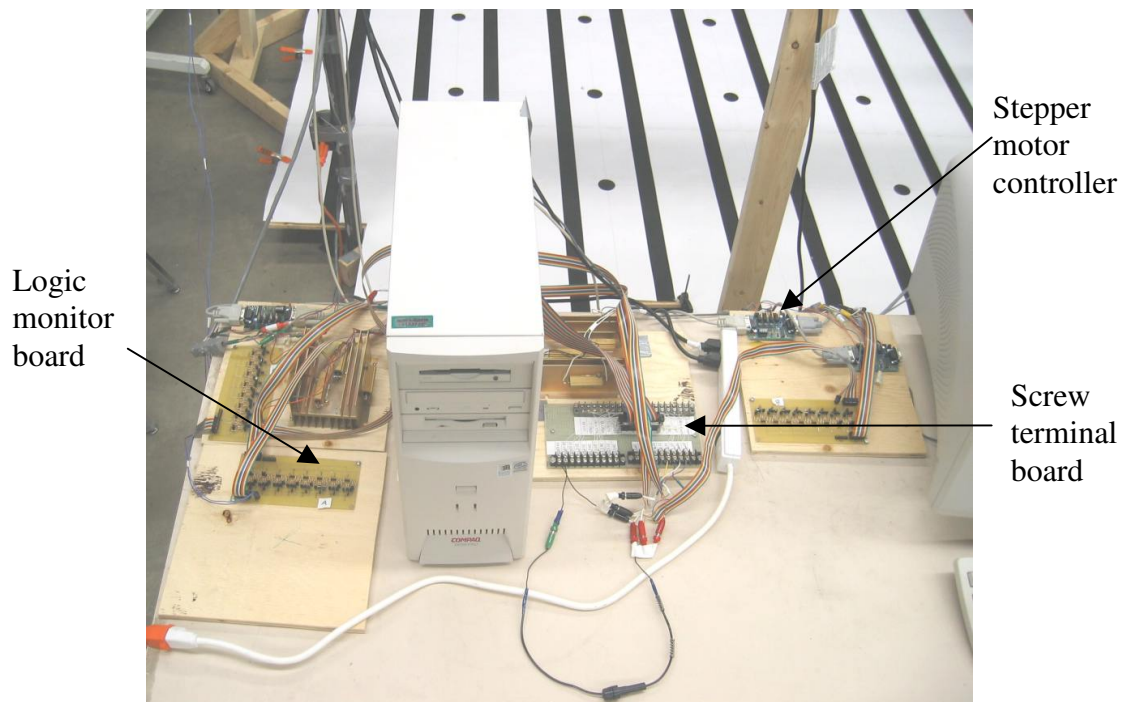


Figure D.5 Control system of the XYZ $\theta$ -table.

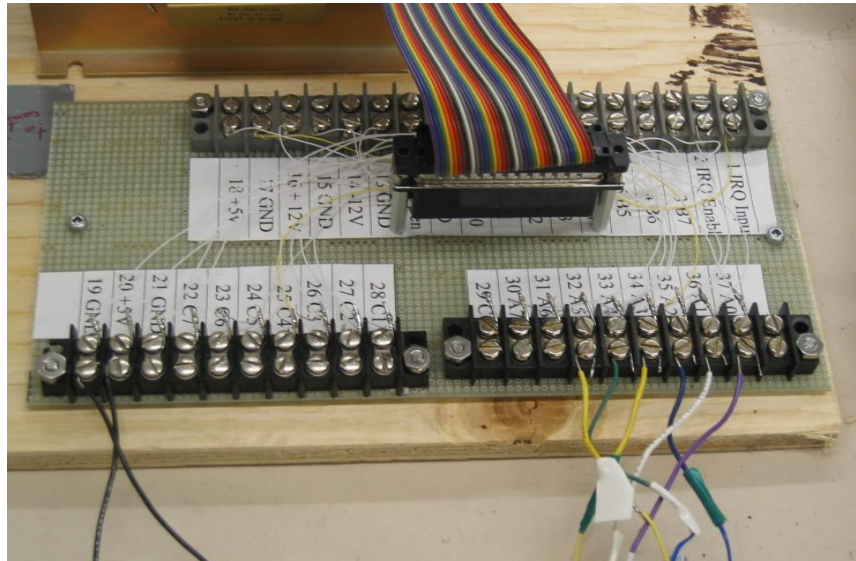


Figure D.6 Screw terminal board.



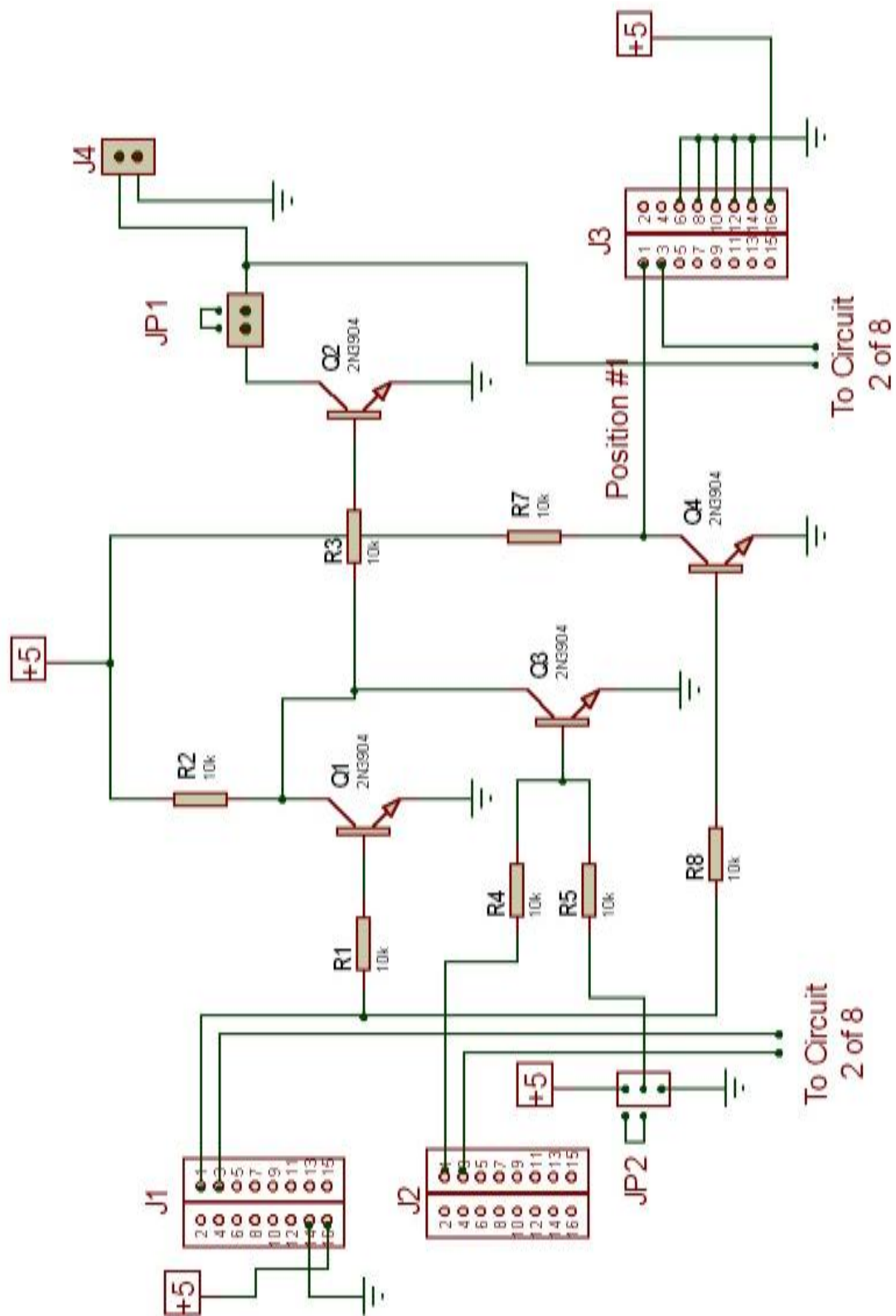


Figure D.7 Simplified circuit (1 of 8) of the logic monitor board.

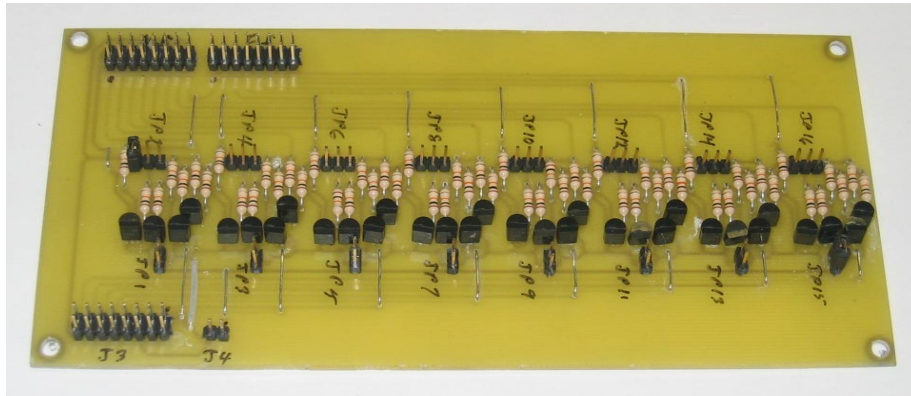


Figure D.8 Picture of a logic monitor board.



Table D.3 Specifications of the controller boards (Weeder Technologies, 2004).

Part name	Stepper Motor Driver Module
Manufacturer	Weeder Technologies, Ft Walton Beach, FL
Drive Type	Quad, open-drain MOSFET
Drive Current	2 A continuous max
Limit Switch	Normally-open, direction sensitive
Idle Current	PWM, selectable in 10% increments
Processor	PIC16CE625
Clock	4 MHz
Communications	9600 Baud, N, 8, 1
Power Requirements	+8 to +30 VDC
Current Draw	9 mA, plus current drawn from motor
Operating Temperature	0° C to 70° C
Board Dimensions	3.1" x 2.0" x 1.0"
Weight	1.9 oz

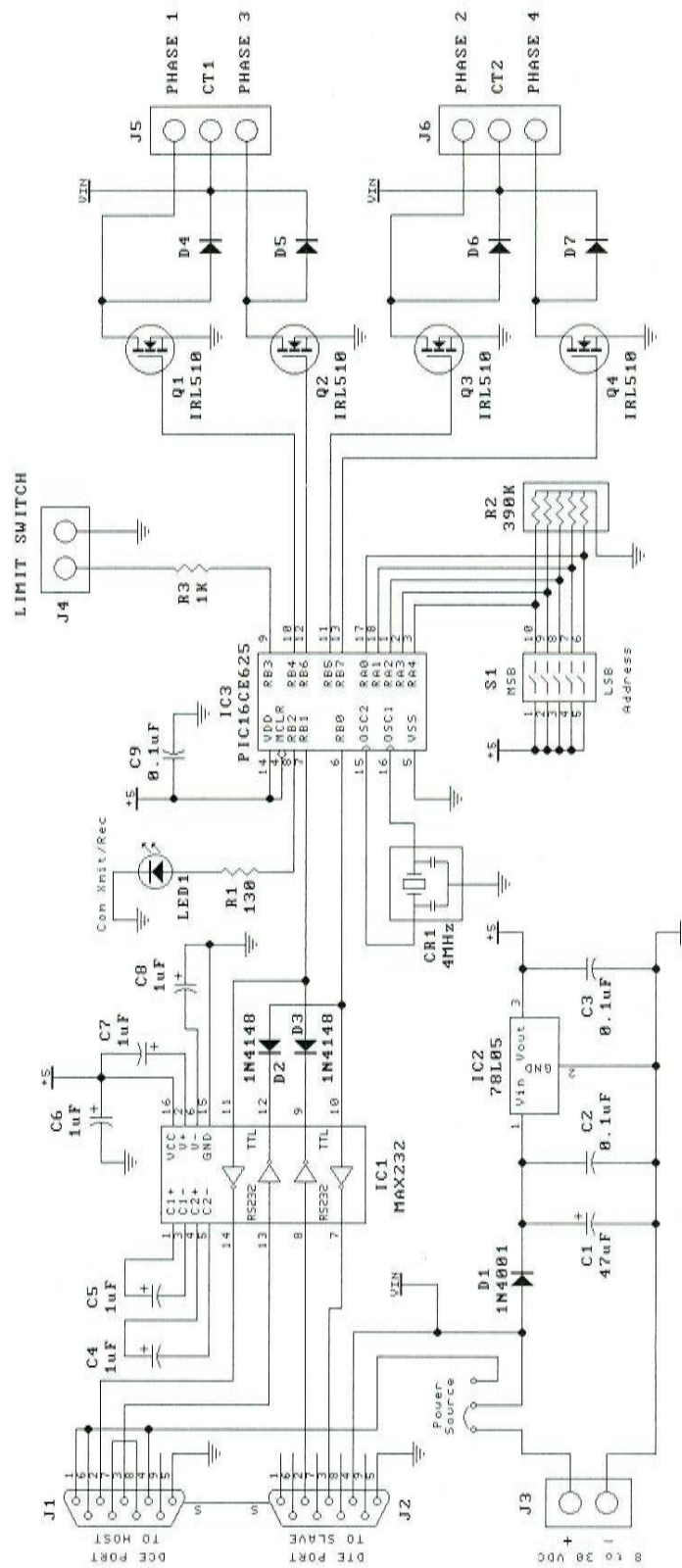


Figure D.9 Circuit of a controller board (Weeder Technologies, 2004).

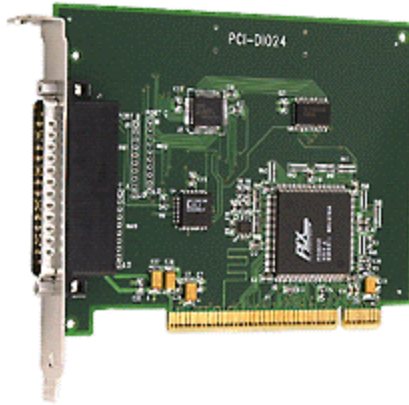


Figure D.10 Digital I/O board (Measurement Computing, 2004)

Table D.4 Specifications of digital I/O board (Measurement Computing, 2004).

Part Name	Digital I/O Board
Part Number	PCI-DIO24
Manufacturer	Measurement Computing, Middleboro, MA
Power Consumption	+5 V Operating, 240 mA typical, 350 mA max
Digital Type	82C55
Configuration	2 banks of 8, 2 banks of 4, programmable by bank as input or output.
Number of Channels	24 I/O
Output High	3.7 V min @ -2.5 mA
Output Low	0.4 V max @ 2.5 mA
Input High	2.2 V min, 5.3 V absolute max
Input Low	0.8 V max, -0.3 V absolute min
Power-Up/Reset State	Input mode (high impedance)
Interrupts	INTA# - mapped to IRQn via PCI BIOS at boot-time
Interrupt Enable	External (IR ENABLE, active low, disabled by default through internal resistor to TTL high) and programmable through PCI9052.  0 = disabled, 1= enabled (default)
Interrupt Sources	External source (IR INPUT), polarity programmable through PCI9052.  1 = active high, 0 = active low (default)
Operating Temperature Range	0° to 50° C
Storage Temperature Range	-20° to 70° C
Humidity	0 to 90% non-condensing

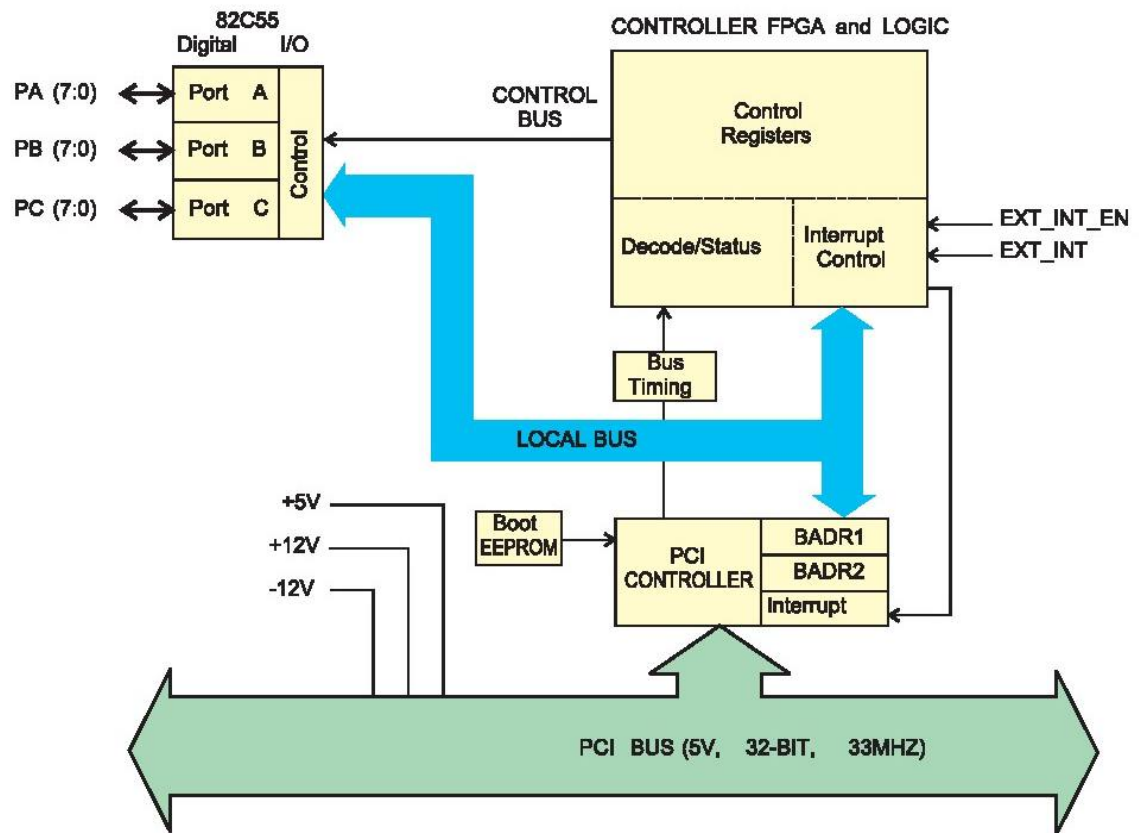


Figure D.11 Block diagram of the digital I/O board (Measurement Computing, 2004).

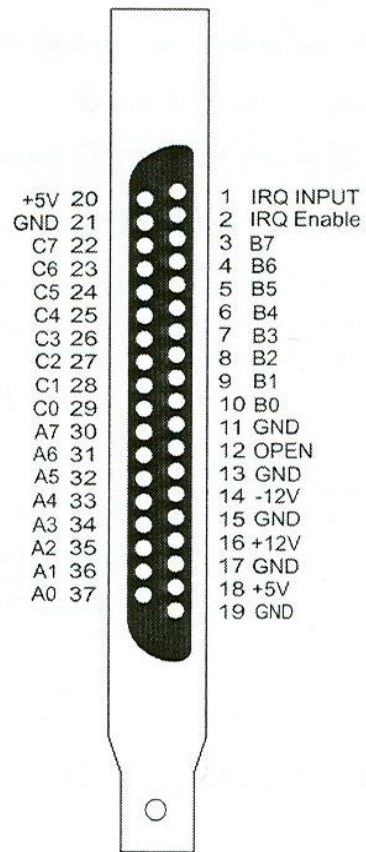


Figure D.12 Pinout of the digital I/O board (Measurement Computing, 2004).

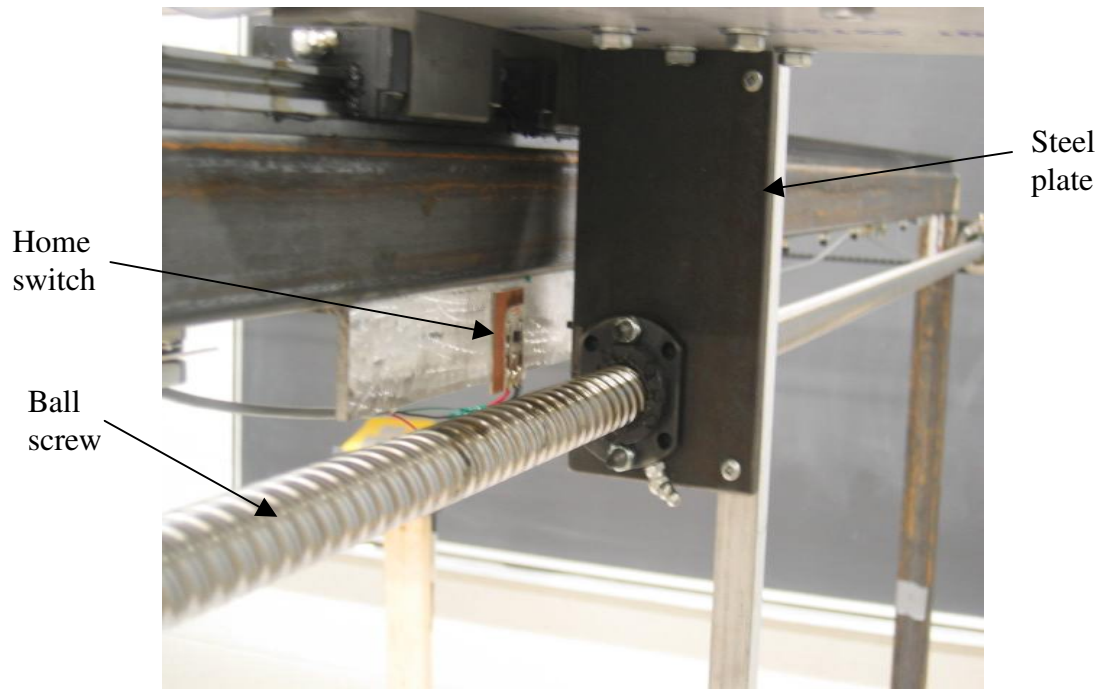


Figure D.13 Hall-effect ‘home’ switch on X-rail.

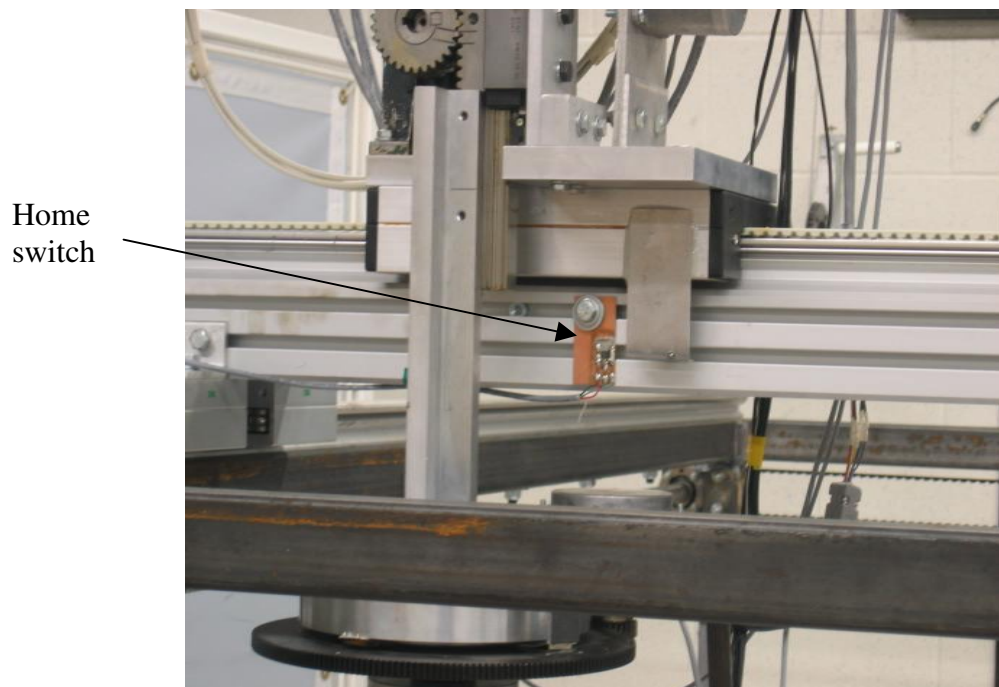


Figure D.14 Hall-effect ‘home’ switch on Y-rail.

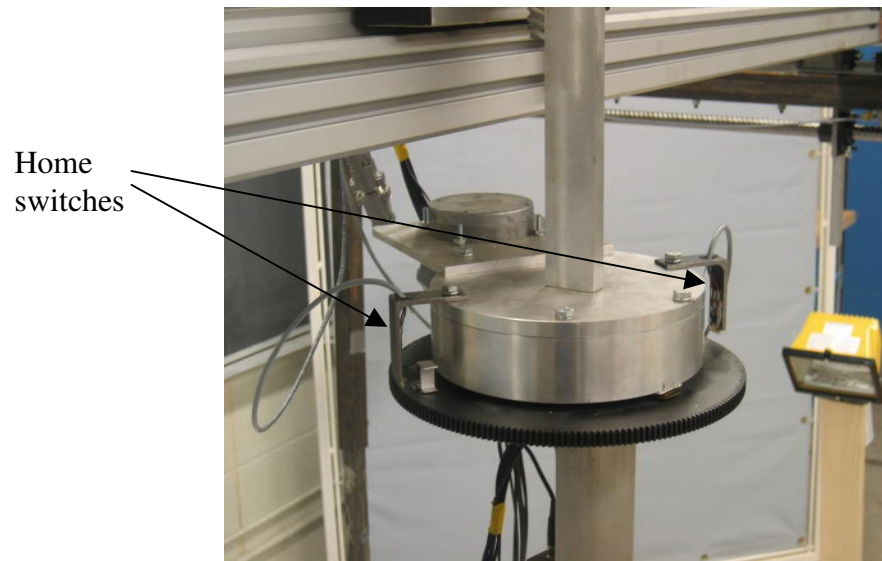


Figure D.15 Hall-effect 'home' switches on  $\theta$ -table.

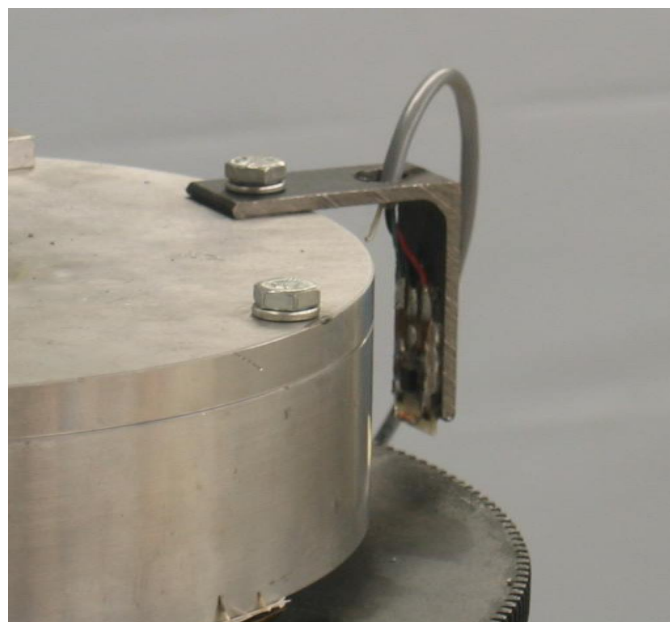


Figure D.16 Close-up of a switch on the  $\theta$ -table.



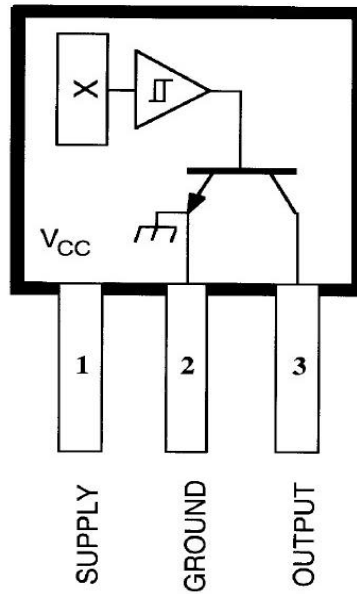


Figure D.17 Circuit of a Hall-effect switch (Allegro Microsystems Inc., 2004).

Table D.5 Specifications of Hall switches (Allegro Microsystems Inc., 2004).

Part Name	Hall-Effect Switch
Part Number	UGN3140U228
Manufacturer	Allegro Microsystems Inc., Worcester, MA
Supply Voltage	25 V
Magnetic Flux Density	Unlimited
Output OFF Voltage	25 V
Continuous Output Current	25 mA
Operating Temperature Range	-20° C to 85° C
Storage Temperature Range	-65° C to 150° C

### D.3 Controller program

This section presents the code and interface of the XYZ $\theta$ -table Controller program. The Control window was presented in Figure 4.18, whereas the other windows included in the interface of the Controller program are presented in Figures D.18 to D.23. To see the code of the Controller program, refer to the file ControlXYZtable.vbp in the folder \Appendix D.3\VB XYZ-table Project\ included on the accompanying CD-ROM. This file was created using Microsoft Visual Basic 6.0 (Microsoft Corporation, Redmond, WA). To facilitate comprehension, comments were inserted in the code. It must be reminded that the program ControlXYZtable.vbp can't be ran without the appropriate framegrabber (Meteor RGB, Matrox Electronics Ltd, Dorval, QC) and digital I/O board (PCI-DIO24, Measurement Computing, Middleboro, MA) being installed on the computer.

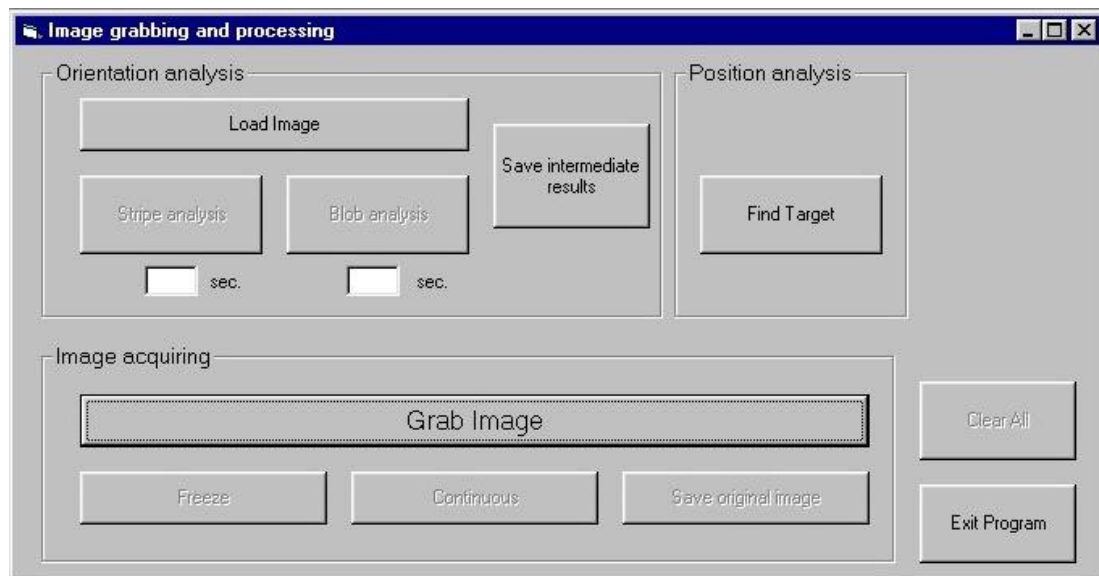


Figure D.18 Image Grabbing and Processing window.

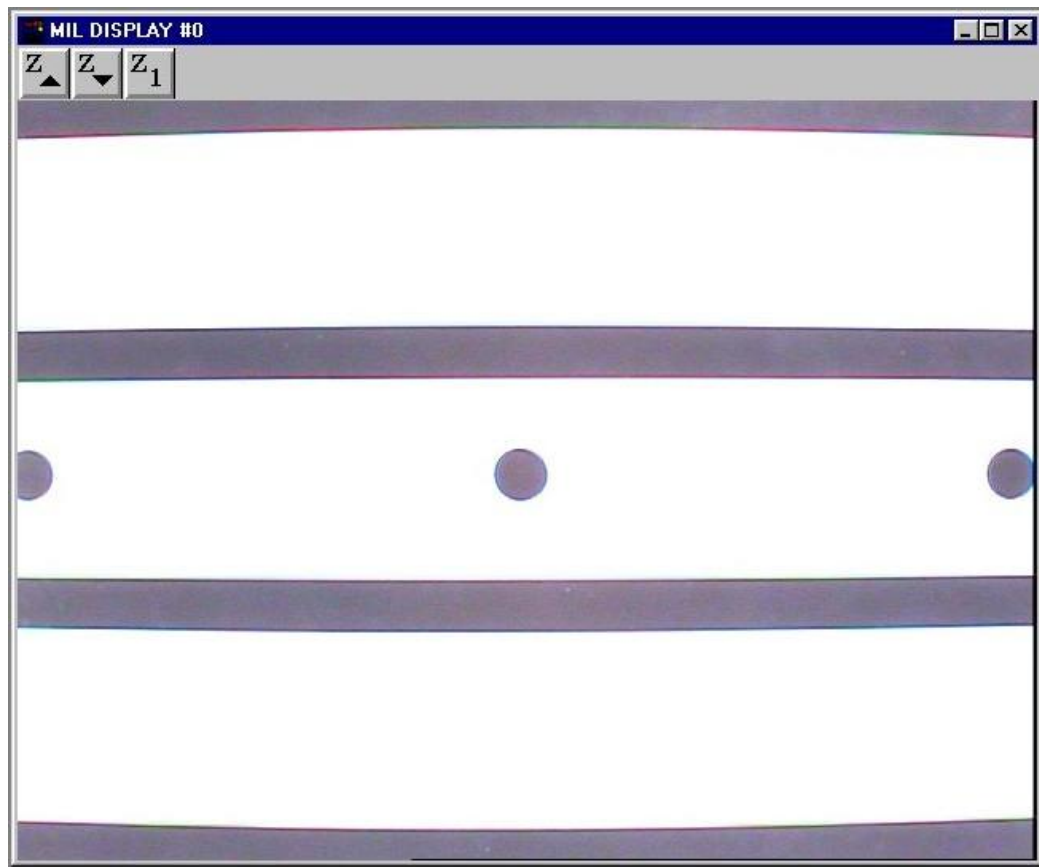


Figure D.19 Image Display window.

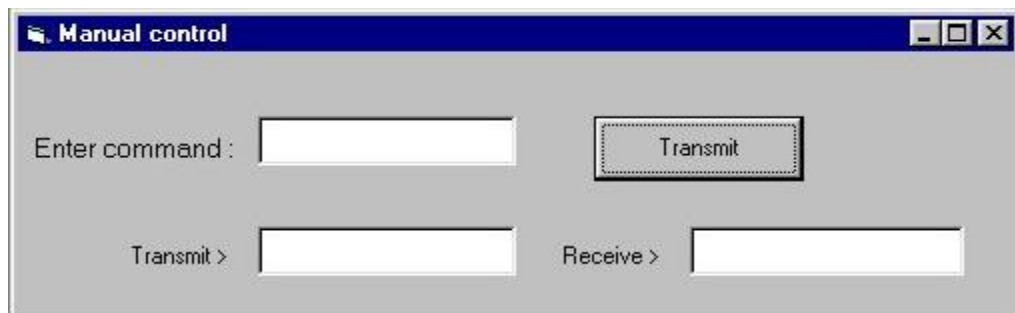


Figure D.20 Manual Control window.

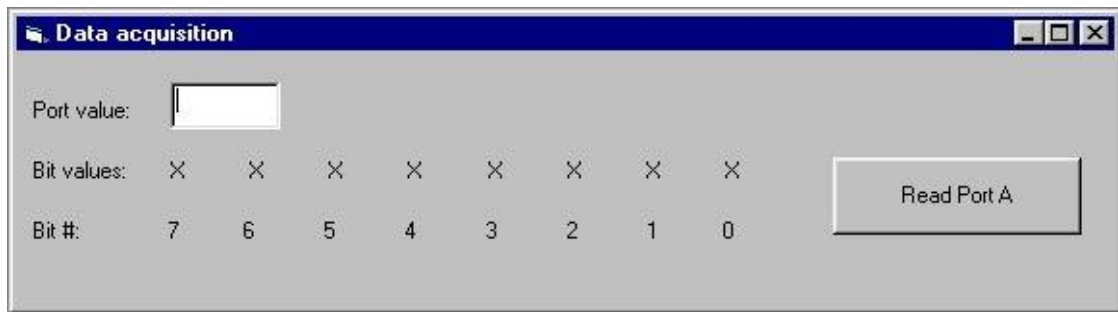


Figure D.21 Data Acquisition window.

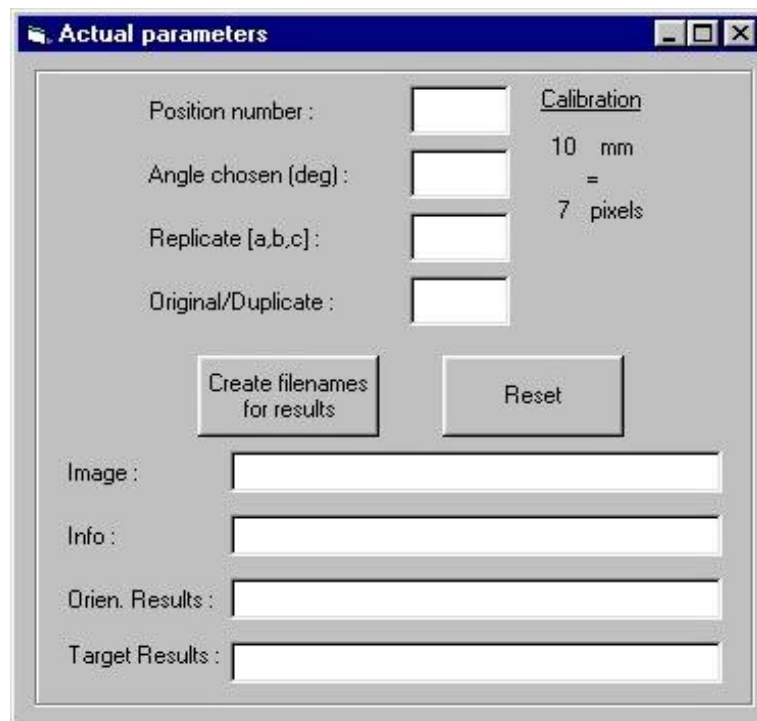


Figure D.22 Actual Parameters window.

The image shows a software window titled "Results". It contains three columns of input fields labeled "Stripe", "Blob", and "Target".

	Stripe	Blob	Target
Angle of rows/interrows (degrees) :	<input type="text"/>	<input type="text"/>	Number of blobs found : <input type="text"/>
Displacement (mm) :	<input type="text"/>	<input type="text"/>	Closest blob (X,Y) (pixels) : ( <input type="text"/> , <input type="text"/> )
Angle of displacement (degrees) :	<input type="text"/>	<input type="text"/>	

At the bottom of the window, there are two buttons: "Save results" and "Close".

Figure D.23 Results window.

## **APPENDIX E      DATA**

The following section presents the original and processed data of the testing of the Line-detection algorithms and of the XYZ $\theta$ -table, as well as the statistical analyses programs and outputs.

### **E.1      Original and processed data**

This section reports the location of the data files on the accompanying CD-ROM.

#### **E.1.1   Line-detection algorithms data**

The images gathered to test the Line-detection algorithms are reported on the accompanying CD-ROM, in the folder \Appendix E\Appendix E.1\Appendix E.1.1\ . The noise-free images are in the file Line-Detection Non-Weedy Images.zip, and the noisy images are in Line-Detection Weedy Images.zip. Each image is labeled as: Image category (O = Original noise-free, D = Duplicate noise-free, OW = Original noisy, DW = Duplicate noisy) - Row width (in inches) - Row spacing (in inches) - angle of the stripes (in deg) - offset ([0,1,2,3]/4 of the Row spacing) - replicate (a, b or c). Ex: OW-2-6-120-3-b.tif. The image files can be viewed with any image processing software.

The file Line-Detection\_Results.xls presents the error on location of the inter-row space, the error in the determination of the rows angle, and the processing time for each image.

### **E.1.2 XYZ $\theta$ -table data**

The images gathered to test the XYZ $\theta$ -table are reported on the accompanying CD-ROM, in the file \Appendix E\Appendix E.1\Appendix E.1.2\Images XYZ-table.zip. The image filenames are labeled as: O - target number (1 to 16) - angle of rotation (deg) - replicate (a,b,c). Ex: O-5-60-c.tif. The image files can be viewed with any image processing software.

The file XYZtableRawData.xls contains the results of the Controller program for each image, i.e. the orientation of the camera and the location of the target in the image.

The file XYZtableMeanLocationCamera.xls contains the source data for Figure 5.16, i.e. the mean location of camera relative to targets for each set of 3 replicates.

The file XYZtableErrorXYrails.xls contains the absolute positional error along the X-axis and the Y-axis for each image.

## E.2 Statistical analyses

This section presents examples of the SAS programs and outputs applied to the data in the Line-detection algorithms testing and the XYZ $\theta$ -table testing. Complete programs and outputs are on the accompanying CD-ROM. The files were created using SAS v.8.02 (SAS Institute Inc., Cary, NC).

### E.2.1 Line-detection algorithms analyses

The following is an example of a SAS program analyzing the error in the determination of the stripes orientation for all algorithms, in the complete set of noise-free images ( $n = 288$ ). The program performed multiple comparisons of the means for each algorithm using the conventional parametrical ANOVA, applied tests for normality, and performed multiple comparisons again in a non-parametrical analysis. This program can be viewed in the file \Appendix E\Appendix E.2\Appendix E.2.1\Error on row angle\AlgosErrA\_Ori\_GLM.sas on the CD-ROM.

\*\*\*\*\*

SAS program

\*\*\*\*\*

```
options ls=75 ps=1000;
```

```
data ori;
```

```
  infile ' H:\SAS\SAS Feb 2004\AlgosErrA_Original.txt' ;  
  input RW RS Angle Offset Rep$ Algo$ ErrA;
```

```
proc print data=ori;
```

```
  title ' Data from non-weedy images (all algos) - Error on AngleRow (deg)' ;
```

```
proc glm;
```

```
  class Algo;  
  model ErrA = Algo/ss3;  
  means Algo/lsd;
```



```

lsmeans Algo/stderr pdiff;
output out=exitGLM r=residGLM p=predGLM;

proc plot data=exitGLM;
  plot residGLM*predGLM/hpos=60 vpos=20;
  plot residGLM*Algo/hpos=60 vpos=20;
run;

proc univariate plot normal data=exitGLM;
  var residGLM;
run;

/*Non-parametrical analysis*/
proc rank data=ori out=nonpara;
  var ErrA;
  ranks rErrA;
run;

proc glm data=nonpara;
  class Algo;
  model rErrA = Algo/ss3;
  means Algo/lsd;
  lsmeans Algo/stderr pdiff;
run;
*****

```

The SAS program yielded the following results (complete output in file \Appendix E\Appendix E.2\Appendix E.2.1\Error on row angle\ AlgosErrA\_Ori\_GLM.lst).

\*\*\*\*\*

## SAS output

\*\*\*\*\*

### The GLM Procedure

Dependent Variable: ErrA

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	1.4526063	0.4842021	3.32	0.0192
Error	1148	167.3996257	0.1458185		
Corrected Total	1151	168.8522319			

R-Square	Coeff Var	Root MSE	ErrA Mean
0.008603	74.09299	0.381862	0.515382

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Algo	3	1.45260625	0.48420208	3.32	0.0192
Data from non-weedy images (all algos) - Error on AngleRow (deg)	253				
21:08 Sunday, February 8, 2004					

### The GLM Procedure

#### t Tests (LSD) for ErrA

NOTE: This test controls the Type I comparisonwise error rate, not the experimentwise error rate.

Alpha	0.05
Error Degrees of Freedom	1148
Error Mean Square	0.145818
Critical Value of t	1.96203
Least Significant Difference	0.0624

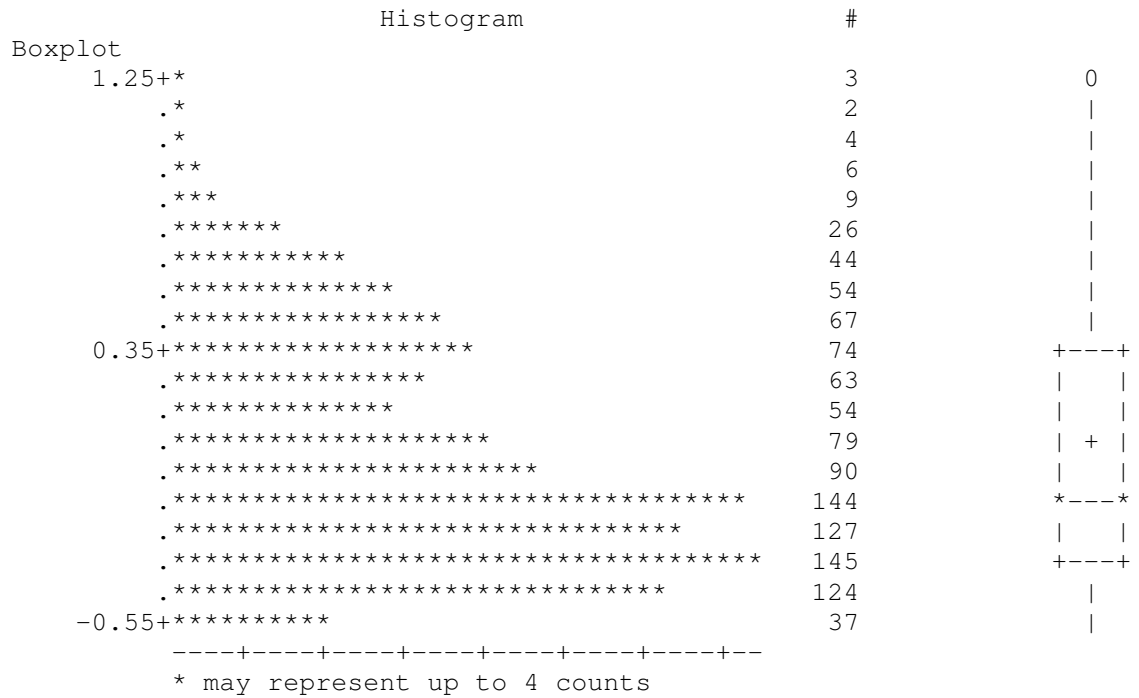
Means with the same letter are not significantly different.

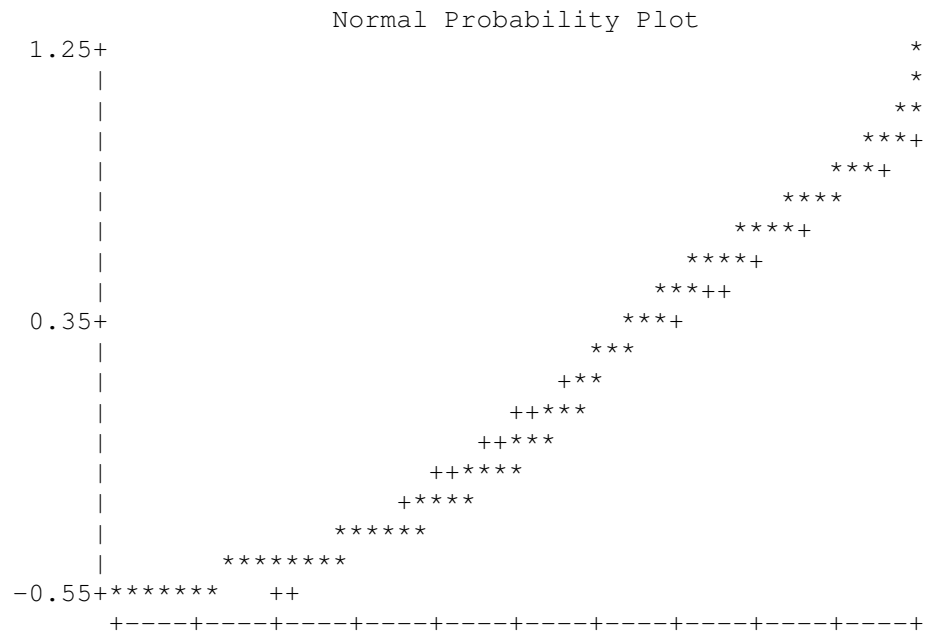
t Grouping	Mean	N	Algo
A	0.56222	288	Reg
A			

B	A	0.53497	288	Stripe
B	C	0.49358	288	Hough
	C	0.47076	288	Blob

## Tests for Normality

Test	--Statistic--	-----p Value-----
Shapiro-Wilk	W 0.944507	Pr < W <0.0001
Kolmogorov-Smirnov	D 0.112966	Pr > D <0.0100
Cramer-von Mises	W-Sq 3.411581	Pr > W-Sq <0.0050
Anderson-Darling	A-Sq 19.87388	Pr > A-Sq <0.0050





#### The GLM Procedure

##### Class Level Information

Class	Levels	Values
Algo	4	Blob Hough Reg Stripe

Number of observations      1152

Data from non-weedy images (all algos) - Error on AngleRow (deg)      259

21:08 Sunday, February 8, 2004

#### The GLM Procedure

Dependent Variable: rErrA      Rank for Variable ErrA

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	3	1821524.6	607174.9	5.55	0.0009
Error	1148	125484640.4	109307.2		
Corrected Total	1151	127306165.0			

R-Square	Coeff Var	Root MSE	rErrA Mean
0.014308	57.34889	330.6164	576.5000

Source	DF	Type III SS	Mean Square	F Value	Pr > F
Algo	3	1821524.620	607174.873	5.55	0.0009
Data from non-weedy images (all algos) - Error on AngleRow (deg)	260				

21:08 Sunday, February 8, 2004

The GLM Procedure

t Tests (LSD) for rErrA

NOTE: This test controls the Type I comparisonwise error rate, not the experimentwise error rate.

Alpha	0.05
Error Degrees of Freedom	1148
Error Mean Square	109307.2
Critical Value of t	1.96203
Least Significant Difference	54.057

Means with the same letter are not significantly different.

t Grouping	Mean	N	Algo
A	626.66	288	Reg
A			
A	604.27	288	Stripe
B	538.25	288	Hough
B			
B	536.82	288	Blob

\*\*\*\*\*

The same analysis was performed on the noisy data set ( $n = 288$ ). The files \Appendix E\Appendix E.2\Appendix E.2.1\Error on row angle\AlgosErrA\_Weedy\_GLM.sas and .lst contain the SAS program and output of that analysis.

The analysis on the noise-free and nosy data sets was performed again without the outliers. The files concerned are included in the same directory and labelled as:

AlgosErrA\_Ori\_GLM\_Outliers.sas and .lst

AlgosErrA\_Weedy\_GLM\_Outliers.sas and .lst.

The outliers were identified by performing an outlier analysis in each noisy and noise-free sub data sets, for each algorithm. In addition to that, the programs performed multiple comparisons of the means to compare the effect of the images parameters on the results using the conventional parametrical ANOVA, applied tests for normality, and performed multiple comparisons again in a non-parametrical analysis if necessary. Each file is labelled as:

[Algorithm]ErrA\_[Data Set]\_GLM.sas for programs

and

[Algorithm]ErrA\_[Data Set]\_GLM.lst for outputs

where:

[Algorithm] = Stripe, Blob, Reg or Hough

[Data Set] = Original (noise-free) or Weedy (noisy)

The multiple comparisons comparing the effect of the images parameters were performed again for each algorithm without the outliers. The files are labelled as:

[Algorithm]ErrA\_[Data Set]\_GLM\_Outliers.sas for programs

and

[Algorithm]ErrA\_[Data Set]\_GLM\_Outliers.lst for outputs

where:

[Algorithm] = Stripe, Blob, Reg or Hough

[Data Set] = Original (noise-free) or Weedy (noisy).

The same analyses were performed on the results of the determination of the inter-row space location. The name of those files follow the same template as the files related to the stripe orientation, except that “ErrA” is replaced by “MinDist” in each filename. The files are located in the folder \Appendix E\Appendix E.2\Appendix E.2.1\Error on inter-row location\ on the accompanying CD-ROM. Ex: AlgoMinDist\_Ori\_GLM.sas.

Finally, the folder \Appendix E\Appendix E.2\Appendix E.2.1\Processing time\ contains files reporting the processing time for each image. The files also include the mean and standard deviation of each data set. The files are labelled as:

[Algorithm]ProcTime\_[Data Set].xls

where:

[Algorithm] = Stripe, Blob, Reg or Hough

[Data Set] = Original (noise-free) or Weedy (noisy).

## E.2.2 XYZ $\theta$ -table analyses

The following is an example of a SAS program analyzing the error in camera positioning. The program evaluated the relationship between the parameters describing the images (target number and rotation angle), then calculated the mean positional error for each set of 3 replicates, at each target and by angle of rotation. Tests for normality were applied on the data (results not shown in the output presented), and a non-parametrical ANOVA was performed again. This program is in the file \Appendix E\Appendix E.2\Appendix E.2.2\XYZTargetTotalmmGLM.sas on the CD-ROM.

\*\*\*\*\*

SAS program

\*\*\*\*\*

```
options ls=75 ps=1000;
```

```
data ori;
```

```
    infile ' H:\SAS\SAS Nov 2003\XYZTable
Testing\Target\XYZTargetSAS4_Totalmm.txt' ;
    input PosNo AChosen ErrPosMM;
```

```

proc print data=ori;
    title ' Data from XYZ table testing - Camera positioning error (mm)' ;
run;

/*Variance analysis and hypo testing*/
proc glm;
    title ' Data from XYZ table testing - Camera positioning error (mm)' ;
    class PosNo AChosen;
    model ErrPosMM = PosNo AChosen PosNo*AChosen/ss3;
    means PosNo*AChosen;
    output out=exitGLM r=residGLM p=predGLM;
run;

proc plot data=exitGLM;
    plot residGLM*predGLM/hpos=60 vpos=20;
run;

proc univariate plot normal data=exitGLM;
    var residGLM;
run;

proc rank data=ori out=nonpara;
    var ErrPosMM;
    ranks rErrPosMM;
run;

proc glm;
    class PosNo AChosen;
    model rErrPosMM=PosNo AChosen PosNo*AChosen/ss3;
run;
*****

```

The program yielded the following results (complete output in \Appendix E\Appendix E.2\Appendix E.2.2\XYZTargetTotalmmGLM.lst).



\*\*\*\*\*

## SAS output

\*\*\*\*\*

### The GLM Procedure

Dependent Variable: ErrPosMM

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	95	3043.052165	32.032128	1173.46	<.0001
Error	192	5.241067	0.027297		
Corrected Total	287	3048.293232			

R-Square	Coeff Var	Root MSE	ErrPosMM Mean
0.998281	2.464393	0.165219	6.704236

Source	DF	Type III SS	Mean Square	F Value	Pr > F
PosNo	15	1855.526821	123.701788	4531.66	<.0001
AChosen	5	1123.476078	224.695216	8231.43	<.0001
PosNo*AChosen	75	64.049267	0.853990	31.28	<.0001

Data from XYZ table testing - Camera positioning error (mm) 109  
09:02 Monday, February 9, 2004

### The GLM Procedure

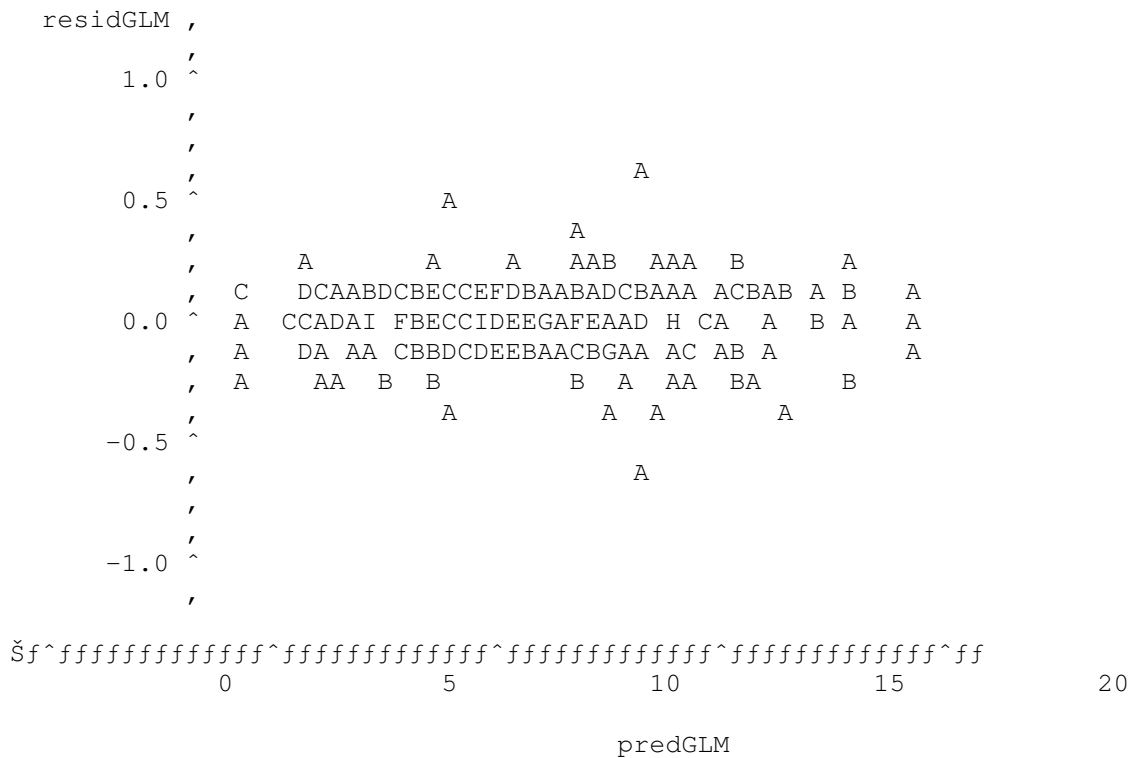
Level of PosNo	Level of AChosen	N	-----ErrPosMM----- Mean	Std Dev
1	0	3	1.8900000	0.15588457
1	30	3	3.2866667	0.08144528
1	60	3	4.2033333	0.15631165
1	90	3	4.9466667	0.17559423
1	120	3	5.3566667	0.06027714
1	150	3	5.6866667	0.05686241
2	0	3	0.4333333	0.19347696
2	30	3	2.1633333	0.14224392
2	60	3	3.4866667	0.23965253
2	90	3	4.8233333	0.07505553
2	120	3	5.7866667	0.14571662
2	150	3	6.2233333	0.08082904
3	0	3	1.6533333	0.21385353
3	30	3	2.4600000	0.06000000
3	60	3	4.0700000	0.15716234
3	90	3	5.6333333	0.11718931
3	120	3	6.9166667	0.08621678

3	150	3	7.7800000	0.07810250
4	0	3	2.7833333	0.08736895
4	30	3	3.1433333	0.04041452
4	60	3	4.6466667	0.26083200
4	90	3	6.4000000	0.06082763
4	120	3	7.8233333	0.28728615
4	150	3	8.9233333	0.10969655
5	0	3	0.2600000	0.12124356
5	30	3	1.5633333	0.04041452
5	60	3	3.2833333	0.04041452
5	90	3	4.5500000	0.08185353
5	120	3	5.6900000	0.05291503
5	150	3	6.2000000	0.12288206
6	0	3	1.9566667	0.11015141
6	30	3	1.7233333	0.05859465
6	60	3	3.2466667	0.14977761
6	90	3	4.8300000	0.10440307
6	120	3	6.3266667	0.12342339
6	150	3	7.4066667	0.09451631
7	0	3	3.9166667	0.08504901
7	30	3	3.4966667	0.18823744
7	60	3	4.7766667	0.02081666
7	90	3	6.5333333	0.17559423
7	120	3	8.2566667	0.21385353
7	150	3	9.4000000	0.07000000
8	0	3	5.7833333	0.11372481
8	30	3	5.2933333	0.14468356
8	60	3	6.1533333	0.10692677
8	90	3	7.8266667	0.22188586
8	120	3	9.5933333	0.31942657
8	150	3	10.8133333	0.01154701
9	0	3	2.2133333	0.19425070
9	30	3	2.4200000	0.18681542
9	60	3	3.9933333	0.04725816
9	90	3	5.6266667	0.12503333
9	120	3	7.0033333	0.02516611
9	150	3	8.1600000	0.05291503
10	0	3	4.1833333	0.07094599
10	30	3	3.7766667	0.11060440
10	60	3	4.7533333	0.11718931
10	90	3	6.5033333	0.11015141
10	120	3	8.0333333	0.12013881
10	150	3	9.3466667	0.09073772
11	0	3	6.5666667	0.19857828
11	30	3	5.9100000	0.11269428
11	60	3	6.8033333	0.02081666
11	90	3	8.3033333	0.11930353
11	120	3	10.0966667	0.00577350
11	150	3	11.4100000	0.28160256
12	0	3	8.4200000	0.14730920
12	30	3	7.8700000	0.07000000
12	60	3	8.5566667	0.11015141
12	90	3	9.9200000	0.11269428
12	120	3	11.6666667	0.25006666
12	150	3	13.1533333	0.07505553
13	0	3	5.1000000	0.42296572
13	30	3	4.6500000	0.16462078

13	60	3	5.5300000	0.11532563
13	90	3	7.1900000	0.01732051
13	120	3	9.0166667	0.23180452
13	150	3	10.1166667	0.03214550
14	0	3	7.0266667	0.15631165
14	30	3	6.0466667	0.11846237
14	60	3	6.7466667	0.09712535
14	90	3	8.4033333	0.15502688
14	120	3	10.1400000	0.19000000
14	150	3	11.5533333	0.17672955
15	0	3	9.3433333	0.59045180
15	30	3	8.6366667	0.21361960
15	60	3	8.7433333	0.34530180
15	90	3	10.4266667	0.25890796
15	120	3	12.2500000	0.08185353
15	150	3	13.7666667	0.18770544
16	0	3	11.3133333	0.20550750
16	30	3	10.4166667	0.13279056
16	60	3	10.9100000	0.08185353
16	90	3	12.6066667	0.27465129
16	120	3	14.0533333	0.23180452
16	150	3	15.5300000	0.08888194

Data from XYZ table testing - Camera positioning error (mm) 110  
09:02 Monday, February 9, 2004

Plot of residGLM\*predGLM. Legend: A = 1 obs, B = 2 obs, etc.



\*\*\*\*\*

A similar analysis was performed to evaluate the error in camera rotation. The SAS program and output of that analysis are in \Appendix E\Appendix E.2\Appendix E.2.2\XYZOrientationGLM.sas and .lst.

## APPENDIX F TOLERANCE OF Y-RAIL DISPLACEMENT

The Y-rail of the XYZ $\theta$ -table should theoretically move in a parallel manner along the X-rails (Fig. F.1). However, because of the tolerance of specific parts of the XYZ $\theta$ -table, one end of the Y-rail may cover a greater distance than the other (Fig. F.2), therefore causing an error when positioning the camera. Estimating that distance can help analysing the results in camera positioning.

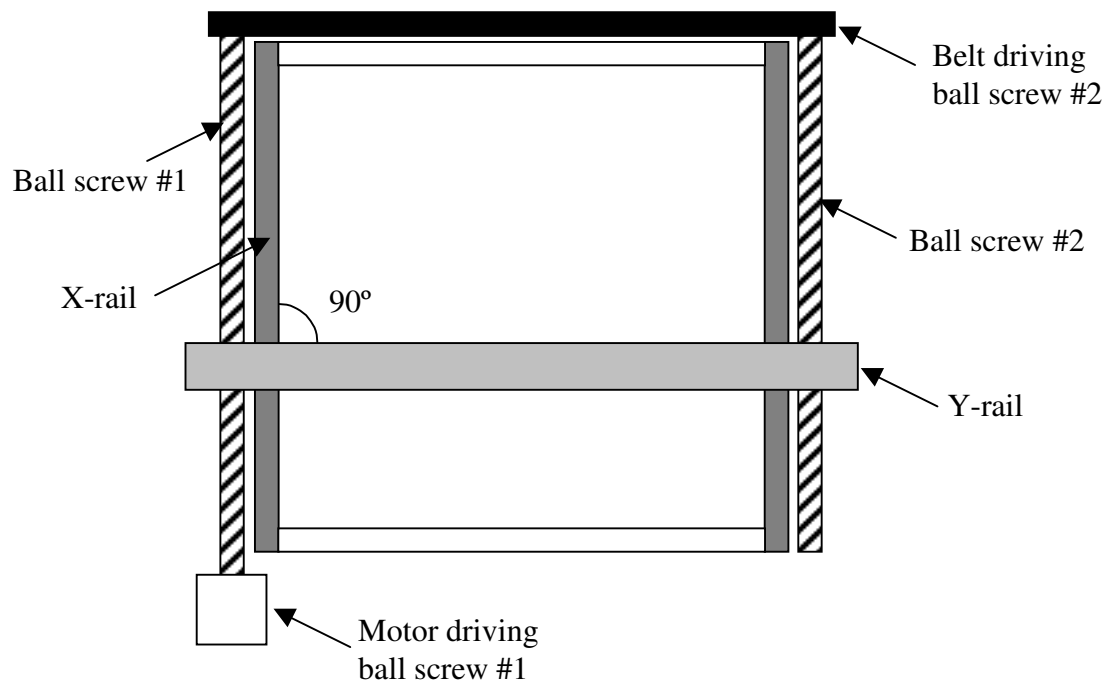


Figure F.1 Theoretical orientation of the Y-rail relative to the X-rails (view from top).

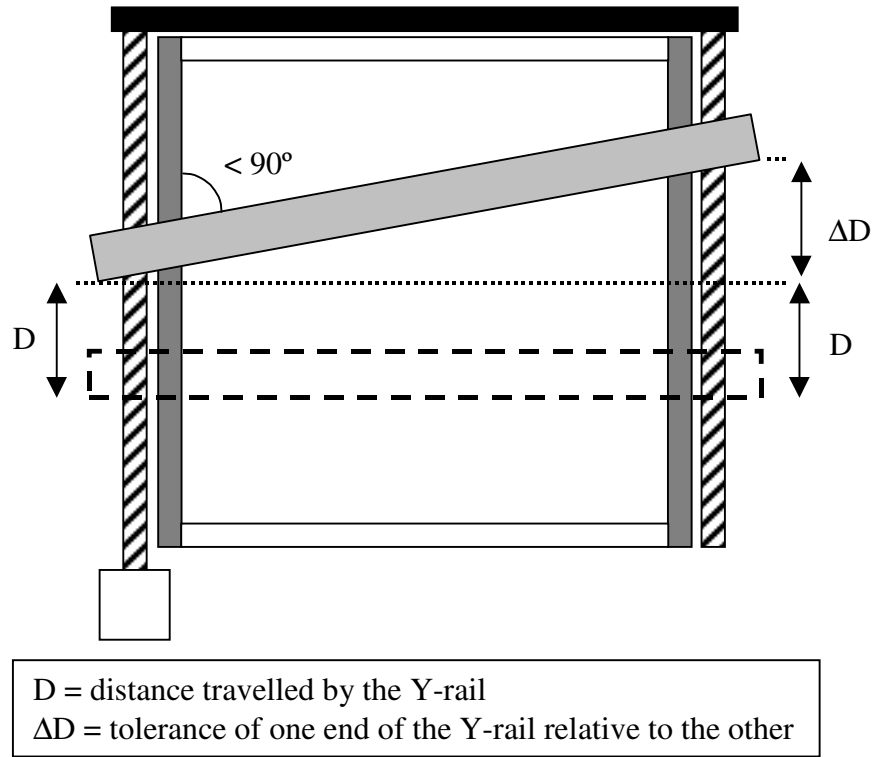


Figure F.2 Tolerance on displacement of Y-rail.

The total tolerance on the displacement of one end of the Y-rail relative to the other is the sum of the tolerance of the two X-rails and the tolerance caused by the timing belt or, as expressed in equation (F.1):

$$\Delta D = 2\Delta X + \Delta B \quad (\text{F.1})$$

where:

$\Delta D$  = tolerance on distance travelled at each end of Y-rail (mm)

$\Delta X$  = tolerance of one X-rail (mm)

$\Delta B$  = tolerance due to the belt timing both ball screws (mm)

The tolerance of each X-rail ( $\Delta X$ ) is influenced by the tolerance of the ball screw and the flexion in the steel plate joining the Y-rail and the ball screw (Fig. D.13). The tolerance of the belt,  $\Delta b$ , can also induce an error in displacement along the X-rail ( $\Delta B$ ), which is given by equation (F.2).

$$\Delta B = p \frac{\Delta b}{2\pi r} \quad (\text{F.2})$$

where:

$\Delta B$  = tolerance along the X-rails due to the timing belt (mm)

$p$  = pitch of the ball screw (mm/rotation)

$\Delta b$  = tolerance of the belt (mm)

$r$  = radius of the pulley (mm)

and,

$2\pi r$  = perimeter of one rotation of the belt pulley (mm/rotation)

Equation (F.1) can then be expressed as:

$$\Delta D = 2\Delta X + p \frac{\Delta b}{2\pi r} \quad (\text{F.3})$$

When estimating the tolerance on the displacement of the camera, the tolerance of other parts of the XYZ $\theta$ -table should be considered, e.g. frame, gear reducers and  $\theta$ -table.